



Red Hat Enterprise Linux OpenStack Platform on HP ConvergedSystem 700x

Table of contents

- Executive summary 2
- Introduction 2
 - About OpenStack 2
 - About RHEL OpenStack Platform 2
 - About HP ConvergedSystem 700x..... 3
- Overview..... 3
 - Intended audience 5
 - Helpful information 5
- Components 5
 - OpenStack architecture 5
 - Reference architecture 8
 - Hardware requirements of ConvergedSystem 700x 8
 - Software requirements 9
 - OpenStack services..... 9
 - Services not covered in this reference architecture 11
 - Supporting technologies 11
 - Deployment model..... 12
- Installation 14
 - HP hardware configuration 14
 - Red Hat OpenStack installation and configuration..... 25
 - Validation 31
- Bill of materials 40
- Implementing a proof-of-concept 40
- Summary 40
- Appendix A: Packstack answer file 41
- Appendix B: Troubleshooting 49
- For more information 50

Executive summary

This paper provides information about our lab implementation of Red Hat® Enterprise Linux® (RHEL) OpenStack Platform 4.0 on HP ConvergedSystem 700x.

OpenStack® makes offering enterprise Infrastructure as a Service (IaaS) Private Cloud a reality. RHEL OpenStack Platform makes implementing and managing OpenStack easier but does not specify hardware deployment or optimization. This white paper includes specific recommendations and best practices for deploying a small but scalable OpenStack cloud on an HP ConvergedSystem 700x system.

HP ConvergedSystem 700x is part of a family of solutions offering simplified, efficient, and reliable application deployment platforms. This solution is built on HP Converged Infrastructure, with integrated and optimized models for RHEL and Red Hat Enterprise Virtualization (RHEV) virtualized workloads. Based on a modular design, ConvergedSystem 700x provides options for components and services to meet a broad set of requirements, deliver seamless scalability and provide an open on-ramp to the cloud.

Target audience: This document is intended for data center administrators, managers, and staff wishing to learn more about this OpenStack on ConvergedSystem 700x deployment. A working knowledge of Linux, SQL databases, DHCP, VLANs, iptables and virtualization is recommended.

Document purpose: The purpose of this document is to describe our lab environment and offer ideas on how you can streamline and optimize your deployment.

This white paper describes testing performed in April 2014.

Introduction

About OpenStack

OpenStack is an open source platform that lets you build an Infrastructure as a Service (IaaS) cloud that runs on commodity hardware. OpenStack is designed for scalability so you can easily add new compute and storage resources to grow your cloud over time. Large organizations such as HP have built massive public clouds on top of OpenStack.

OpenStack is more than a standard software package; it lets you integrate a number of different technologies to construct a cloud. Although the number of options to do this may appear daunting at first, the OpenStack approach provides the greatest amount of flexibility to the users.

About RHEL OpenStack Platform

Red Hat Enterprise Linux OpenStack Platform provides the foundation to build a private or public IaaS cloud on top of Red Hat Enterprise Linux. It offers a massively scalable, fault-tolerant platform for the development of cloud-enabled workloads.

The current Red Hat Enterprise Linux OpenStack Platform 4.0 is based on OpenStack Havana (2013.2 release).

- Fully distributed object storage
- Persistent block-level storage
- Virtual-machine provisioning engine and image storage
- Authentication and authorization mechanism
- Integrated networking
- Web browser-based GUI for both users and administration

The Red Hat Enterprise Linux OpenStack Platform IaaS cloud is implemented by a collection of interacting services that control its computing, storage, and networking resources. The cloud is managed using a web-based interface that allows administrators to control, provision, and automate OpenStack resources. Additionally, the OpenStack infrastructure is facilitated through an extensive API, which is also available to end users of the cloud.

About HP ConvergedSystem 700x

The ConvergedSystem 700x family of solutions offers you simplified and reliable application deployment platforms built on HP Converged Infrastructure. The solutions have a modular architecture and a large array of options to provide access to the cloud, including:

- Accelerated business outcomes with greater simplicity
- Reduced time to value from pre-optimized, complete solutions
- Built-in resource provisioning
- Integrated management
- Single vendor solution lifecycle support
- Reduced risk from superior infrastructure and HP best practices
- Twenty years of innovation and leadership
- Reliable implementation based on proven technology

ConvergedSystem 700x provides standardized building blocks of server, storage, networking, rack and power, and HP innovation. At its core, ConvergedSystem 700x includes:

- HP ProLiant BL460c Gen8 servers in an HP BladeSystem c7000 enclosure with HP Virtual Connect FlexFabric interconnects for the simplest, most cost-efficient virtualization platform (requiring 95 percent fewer cables, NICs and switches than the competition).
- HP 3PAR StoreServ 7000 or 1000, for efficient, flexible and easy-to-manage storage with non-disruptive scaling of capacity and performance (supporting twice as many VMs as the competition).
- HP FlexNetwork high-performance, low-latency architecture ideal for virtualized data centers (enabling 40 percent faster virtual migration than alternative multi-tiered approaches).
- HP options for flexibility and optimization at every level.
- HP and partner services for comprehensive solution support and services offerings, from consulting to delivery to lifecycle support.

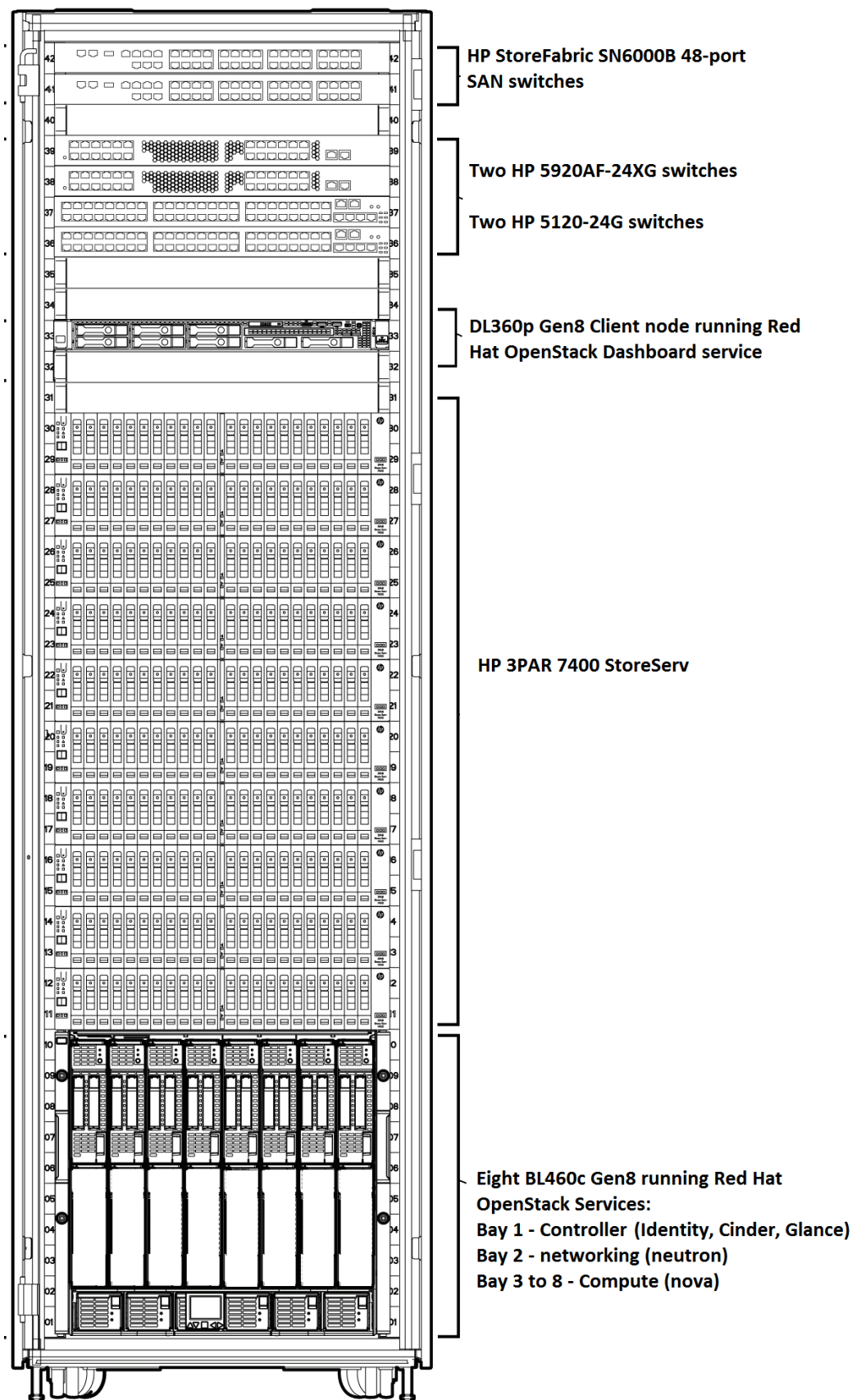
Overview

This white paper has been created to provide guidance in the deployment of a RHEL OpenStack Platform 4.0 cloud on the HP ConvergedSystem 700x.

The ConvergedSystem 700x has been chosen, and we describe the steps necessary to successfully install RHEL OpenStack Platform 4.0 on this hardware providing a small private cloud which may be scaled up by using additional compute nodes. This document presents an architectural view of an RHEL OpenStack Platform private cloud, and describes this as implemented on an HP ConvergedSystem 700x. This document has been written as a companion to the RHEL OpenStack Platform and OpenStack.org documentation for a dual purpose.

1. To examine best practices, deployment, and integration excellence with:
 - Ensured business continuity through ease of deployment and consistent high availability
 - Comprehensive strategies for backup, disaster recovery, and security
 - Greater storage versatility and value
 - Superior networking innovation
 - End-to-end support ownership
2. To examine how to lower costs and provide greater investment protection with:
 - Greater efficiencies from a solution architecture of HP ProLiant servers, HP 3PAR StoreServ arrays, HP FlexNetwork architecture, and comprehensive management
 - Multi-OS, heterogeneous infrastructure support
 - Hardware and software compatibility
 - Easily expandable infrastructure and a flexible on-ramp to the cloud

Figure 1. HP ConvergedSystem 700x as configured for our lab implementation



Intended audience

To be successful with this guide:

- You are familiar with the Red Hat distribution of Linux, SQL databases, and virtualization.
- You are comfortable administering and configuring multiple Linux machines for networking.
- You are comfortable installing and maintaining a MySQL database, and occasionally running SQL queries against it.
- You are familiar with concepts such as DHCP, Linux bridges, VLANs, and iptables.
- You have access to configure the switches and routers.

Helpful information

OpenStack Foundation documentation is available at <http://docs.OpenStack.org>. The *OpenStack Operations Guide* provides invaluable insights and guidance to consider as you design and create your RHEL OpenStack Platform cloud. You can also find information on installation, configuration, training, user guides and even how to develop applications and contribute code.

Additional documentation for the Red Hat Enterprise Linux OpenStack Platform in the Red Hat customer portal is available at: https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform.

The following documents are included:

- **Administration user guide**
How-to procedures for administering Red Hat Enterprise Linux OpenStack Platform environments
- **Configuration reference guide**
Configuration options and sample configuration files for each OpenStack component
- **End user guide**
How-to procedures for using Red Hat Enterprise Linux OpenStack Platform environments
- **Getting started guide**
Packstack deployment procedures for a Red Hat Enterprise Linux OpenStack Platform cloud, as well as brief instructions for getting your cloud up and running
- **Installation and configuration guide**
Deployment procedures for a Red Hat Enterprise Linux OpenStack Platform cloud; procedures for both a manual and foreman installation are included. Also included are brief procedures for validating and monitoring the installation.
- **Release notes**
Information about the current release, including notes about technology previews, recommended practices, and known issues
- **Technical notes**
These Technical Notes are provided to supplement the information contained in the text of Red Hat Enterprise Linux OpenStack Platform errata advisories released through Red Hat Network

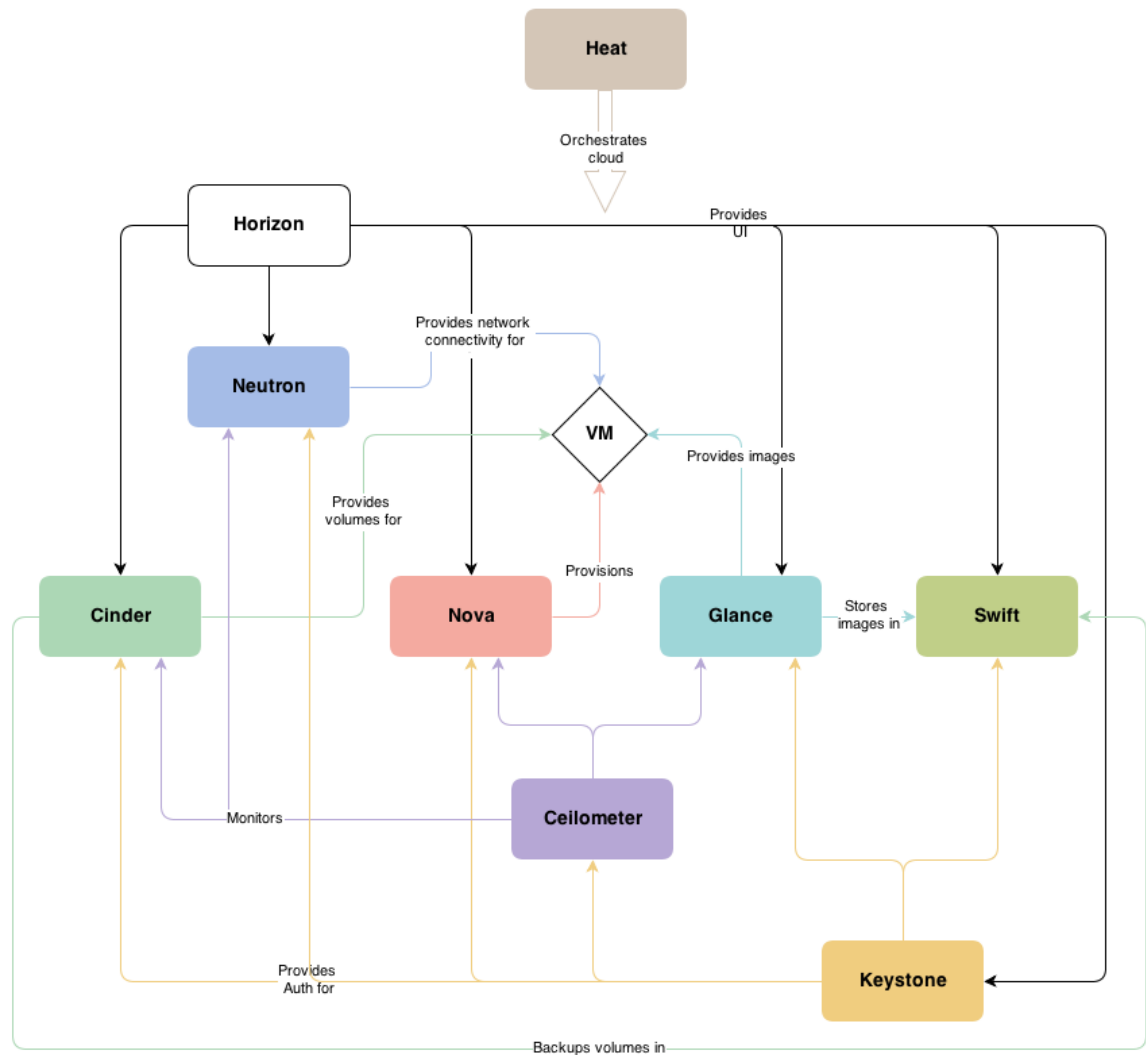
Please download the “OpenStack® HP 3PAR StoreServ Block Storage Drivers Configuration Best Practices” document, available at <http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA4-9853ENW> as we will reference this document later in the deployment.

Other documentation related to configuring your HP servers will be referenced when required.

Components

OpenStack architecture

OpenStack is designed to be massively horizontally scalable, which allows all services to be distributed widely. However, to simplify this guide we have decided to discuss services of a more central nature using the concept of a single cloud controller. As described in this guide, the cloud controller is a single node that hosts the databases, message queue service, authentication and authorization service, image management service, and externally accessible API endpoints for OpenStack services.

Figure 2. OpenStack conceptual architecture**Cloud controller**

The cloud controller provides the central management system for multi-node OpenStack deployments. Typically, the cloud controller manages authentication and sends messages to all the systems through a message queue. For our example, the cloud controller has a collection of nova-* components that represent the global state of the cloud, talk to services such as authentication, maintain information about the cloud in a database, communicate with all compute nodes and storage workers through a queue, and provide API access. Each service running on a designated cloud controller may be broken out into separate nodes for scalability or availability. It's also possible to use virtual machines for all or some of the services that the cloud controller manages, such as the message queuing.

In this reference architecture we used a single cloud controller server to host the OpenStack management services. By doing this we are trading off fault tolerance for simplicity. It's possible to configure a fully redundant and highly available cloud controller configuration by replicating services and clustering the database storage and message queue capability. We have chosen an implementation that runs all services directly on the cloud controller. This provides a simple and scalable configuration that works well for small to medium size clouds.

Database

Most OpenStack Compute central services, and currently also the nova-compute nodes, use the database for stateful information. Loss of database availability leads to errors. As a result, in a production deployment you should consider clustering your databases in some way to make them failure tolerant. The reference architecture explained in this white paper does not implement a clustered database configuration.

Message queue

Most OpenStack Compute services communicate with each other using the Message Queue. In general, if the message queue fails or becomes inaccessible, the cluster grinds to a halt and ends up in a "read only" state, with information stuck at

the point where the last message was sent. In a large production OpenStack environment it is recommended that you cluster the message queue; Qpid has built-in abilities to do this. However, implementation of a clustered message queue is beyond the scope of this white paper.

Scheduler

Fitting various sized virtual machines (different flavors) into different sized physical nova-compute nodes is a challenging problem. To support your scheduling choices, OpenStack Compute provides several different types of scheduling drivers, a full discussion of which is found in the reference manual (http://docs.openstack.org/trunk/openstack-ops/content/cloud_controller_design.html#scheduling). The reference architecture uses the default libvirt-based scheduler with Kernel-based Virtual Machine (KVM) for virtualization.

For availability purposes, or for very large or high-schedule frequency installations, you should consider running multiple nova-scheduler services. No special load balancing is required, as the nova-scheduler communicates entirely using the message queue.

Images

The OpenStack Image Service consists of two parts – glance-api and glance-registry. The former is responsible for the delivery of images; the compute node uses it to download images from the back-end. The latter maintains the metadata information associated with virtual machine images and requires a database.

The glance-api part is an abstraction layer that allows a choice of back-end. Currently, it supports:

- OpenStack Object Storage: Allows you to store images as objects.
- File system: Uses any traditional file system to store the images as files.
- S3: Allows you to fetch images from Amazon S3.
- HTTP: Allows you to fetch images from a web server. You cannot write images by using this mode.

This reference architecture uses HP 3PAR to provide a file system to store images. You can make use of advanced HP 3PAR features for thin provisioning and replication for this file system.

Dashboard

The OpenStack Dashboard is implemented as a Python web application that runs in the Apache web-server (httpd). It is accessed using a web browser via traditional http protocol. Because it uses the service APIs for the other OpenStack components, it must also be able to reach the API servers (including their admin endpoints) over the network.

Authentication and authorization

The concepts supporting OpenStack authentication and authorization are derived from well understood and widely used systems of a similar nature. Users have credentials they can use to authenticate, and they can be a member of one or more groups (known as projects or tenants interchangeably).

For example, a cloud administrator might be able to list all instances in the cloud, whereas a user can only see those in their current group. Resources quotas, such as the number of cores that can be used, disk space, etc., are associated with a project.

The OpenStack Identity Service (Keystone) is the point that provides the authentication decisions and user attribute information, which is then used by the other OpenStack services to perform authorization. Policy is set in the `policy.json` file.

The Identity Service supports different plugins for back-end authentication decisions, and storing information. These range from pure storage choices to external systems, and currently include:

- In-memory Key-Value Store
- SQL database
- PAM
- LDAP

Many deployments use the SQL database; however, LDAP is also a popular choice for those with an existing authentication infrastructure that needs to be integrated. In organizations that have a centralized LDAP server for authentication, using LDAP allows synchronizing its use with the HP Integrated Lights-Out (iLO) based credentials used to access the server iLO management controller so it is a good choice in this case. This reference architecture uses a SQL database for the identity storage instead of depending on LDAP being present. If LDAP is available, the OpenStack Operations Guide shows how you can configure LDAP to enable its use with the OpenStack Identity Service.

Network considerations

Because the cloud controller handles so many different services, it must be able to handle the amount of traffic that hits it. For example, if you choose to host the OpenStack Imaging Service on the cloud controller, the cloud controller should be able to support the transferring of the images at an acceptable speed. We recommend that you use a fast NIC, such as 10 GbE. This reference architecture makes use of 10 GbE network connections via HP Virtual Connect FlexFabric modules.

Reference architecture

When implementing a Red Hat Enterprise Linux OpenStack Platform cloud, you will need to make many choices that influence the resulting implementation. For this document we've made some decisions that allow for a small-to-medium size cloud installation that scales well. In this reference architecture implementation, the following design has been considered:

- One blade server acts as the cloud controller by hosting services including the compute and API services.
- Another blade server acts as the network node by hosting OpenStack Networking (neutron) services.
- All of the other blade servers act as compute nodes by hosting nova services.
- One rack server acts as a client node and also hosts the dashboard services.

We have specified a set of compute nodes with a uniform configuration. Adding additional compute capacity is as simple as adding additional compute nodes. The sections below provide more details on the hardware, software, and procedures used to configure this reference architecture in the lab.

Hardware requirements of ConvergedSystem 700x

Table 1 shows the set of hardware components used for this reference architecture in the lab.

Table 1. ConvergedSystem 700x hardware requirements

Component	Purpose
One HP BladeSystem c7000 enclosure	Enclosure to host blades and Virtual Connect modules
Two Virtual Connect FlexFabric 10 Gb/24-Port Modules	Virtual Connect module for Ethernet and SAN connectivity
Eight ProLiant BL460c Gen8 E5-v2 server blades	Blade Servers to host OpenStack services
One ProLiant DL360p Gen8 E5-v2 management server	Rack Server to act as a Client
One HP 3PAR StoreServ 7400	Storage back-end for Glance Image service and Cinder Block Storage service
Two HP StoreFabric SN6000B 24-port SAN switches	Fibre Channel Switches for SAN connectivity between servers and 3PAR
Two HP 5920AF-24XG switches	10 GbE Top-of-Rack switches
Two HP 5120-24G EI switches	Ethernet switches

Note

For this reference architecture an additional server installed with Microsoft® Windows® Server 2008 R2 operating system was used as a jumpstation. This server was used to download or install any necessary software components, and connect to iLOs, Virtual Connect Manager and Onboard Administrator. HP 3PAR Management Console was installed on this server to manage the HP 3PAR used for this reference architecture.

Software requirements

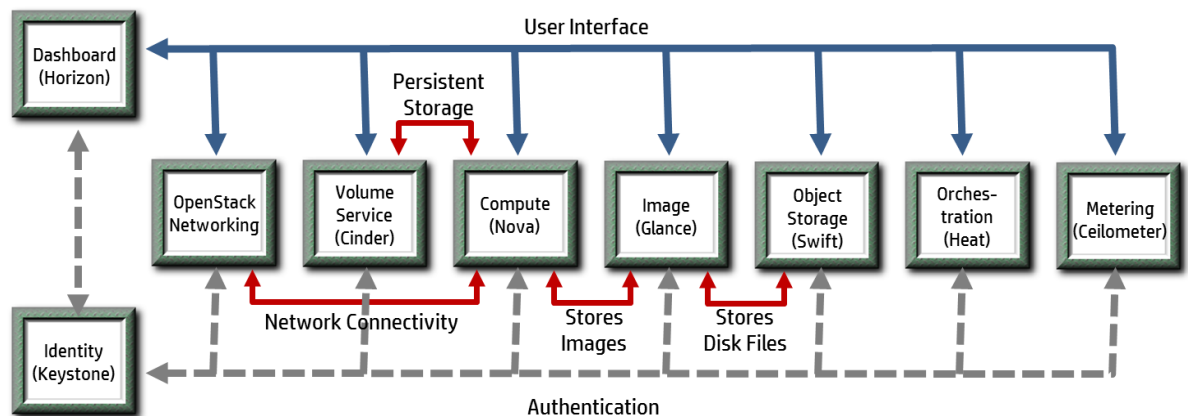
All servers must meet the following software requirements:

- Running Red Hat Enterprise Linux 6.5
- Registered to Red Hat Network (RHN) or the Red Hat Content Delivery Network (CDN)
- Subscribed to following repositories:
 - Red Hat Enterprise Linux 6
 - Red Hat Enterprise Linux OpenStack Platform 4.0

OpenStack services

The image below depicts the RHEL OpenStack Platform services and their interactions with each other.

Figure 3. OpenStack services



Keystone – Identity service

This is a central authentication and authorization mechanism for all OpenStack users and services. It supports multiple forms of authentication including standard username and password credentials, token-based systems and AWS-style logins that use public/private key pairs. It can also integrate with existing directory services such as LDAP.

The Identity service catalog lists all of the services deployed in an OpenStack cloud and manages authentication for them through endpoints. An endpoint is a network address where a service listens for requests. The Identity service provides each OpenStack service – such as Image, Compute, or Block Storage – with one or more endpoints.

The Identity service uses tenants to group or isolate resources. By default, users in one tenant can't access resources in another even if they reside within the same OpenStack cloud deployment or physical host. The Identity service issues tokens to authenticated users. The endpoints validate the token before allowing user access. User accounts are associated with roles that define their access credentials. Multiple users can share the same role within a tenant. The Identity service is comprised of the keystone service, which responds to service requests, places messages in queue, grants access tokens, and updates the state database.

Glance – Image service

This service registers and delivers virtual machine images. They can be copied via snapshot and immediately stored as the basis for new instance deployments. Stored images allow OpenStack users and administrators to provision multiple servers quickly and consistently. The Image Service API provides a standard RESTful interface for querying information about the images.

By default the Image service stores images in the `/var/lib/glance/images` directory of the local server's filesystem where Glance is installed. The Glance API can also be configured to cache images in order to reduce image staging time. The Image service is composed of the `openstack-glance-api` that delivers image information from the registry service and the `openstack-glance-registry` which manages the metadata associated with each image.

Nova – Compute service

OpenStack Compute provisions and manages large networks of virtual machines. It is the backbone of OpenStack's IaaS functionality. OpenStack Compute scales horizontally on standard hardware, enabling the favorable economics of cloud computing. Users and administrators interact with the compute fabric via a web interface and command line tools.

Key features of OpenStack Compute include:

- Distributed and asynchronous architecture, allowing scale out fault tolerance for virtual machine instance management.
- Management of commoditized virtual server resources, where predefined virtual hardware profiles for guests can be assigned to new instances at launch.
- Tenants to separate and control access to compute resources.
- VNC access to instances via web browsers.

OpenStack Compute is composed of many services that work together to provide the full functionality. The `openstack-nova-cert` and `openstack-nova-consoleauth` services handle authorization. The `openstack-nova-api` responds to service requests and the `openstack-nova-scheduler` dispatches the requests to the message queue. The `openstack-nova-conductor` service updates the state database which limits direct access to the state database by compute nodes for increased security. The `openstacknova-compute` service creates and terminates virtual machine instances on the compute nodes. Finally, `openstack-nova-novncproxy` provides a VNC proxy for console access to virtual machines via a standard web browser.

Cinder – Block Storage service

While the OpenStack Compute service provisions ephemeral storage for deployed instances based on their hardware profiles, the OpenStack Block Storage service provides compute instances with persistent block storage. Block storage is appropriate for performance sensitive scenarios such as databases or frequently accessed file systems. Persistent block storage can survive instance termination. It can also be moved between instances like any external storage device. This service can be backed by a variety of enterprise storage platforms or simple NFS servers. This service's features include:

- Persistent block storage devices for compute instances
- Self-service volume creation, attachment, and deletion
- A unified interface for numerous storage platforms
- Volume snapshots

The Block Storage service is comprised of `openstack-cinder-api` which responds to service requests and `openstack-cinder-scheduler` which assigns tasks to the queue. The `openstack-cinder-volume` service interacts with various storage providers to allocate block storage for virtual machines. By default the Block Storage server shares local storage via the iSCSI `tgtd` daemon.

Neutron – Network service

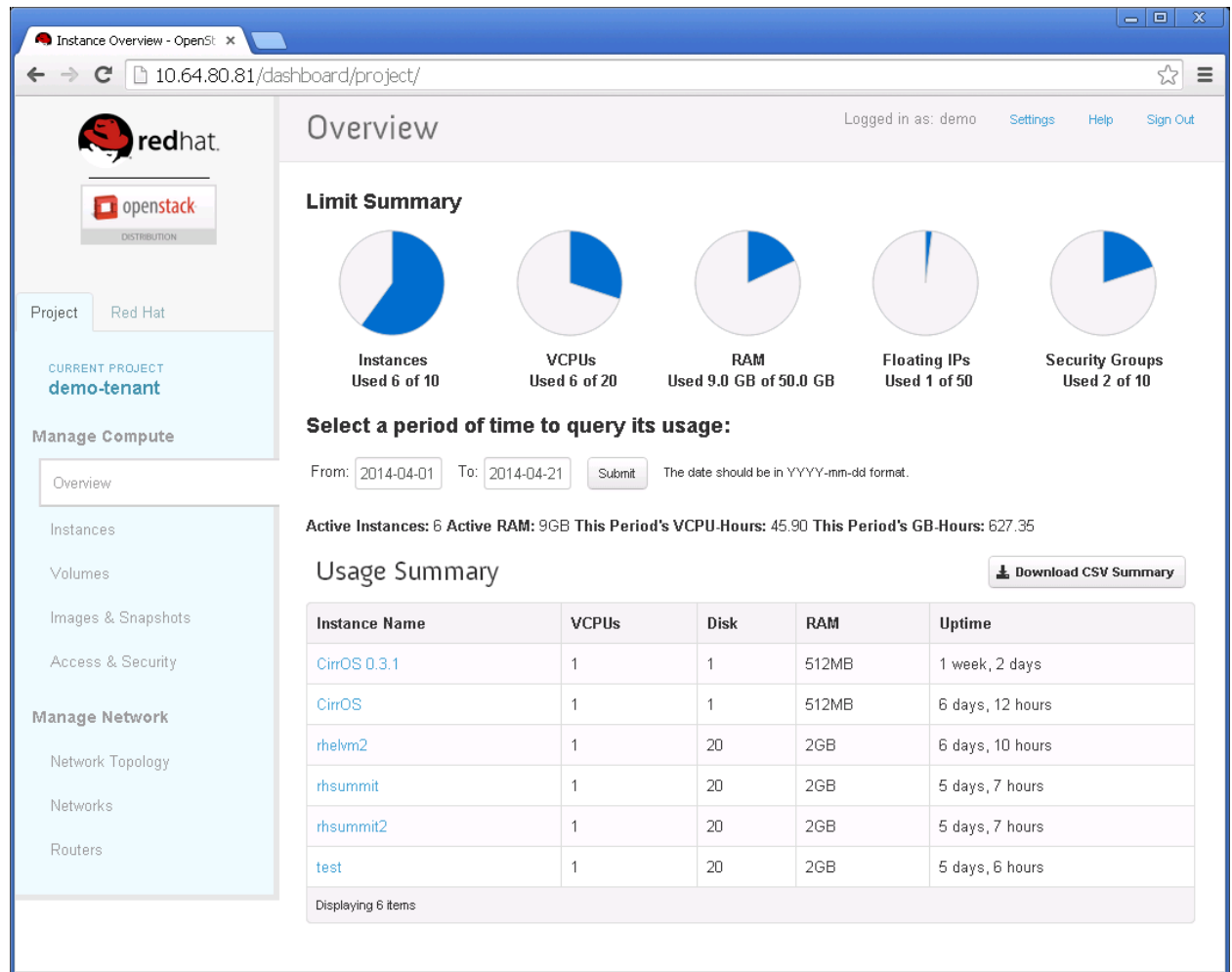
OpenStack Networking is a scalable API-driven service for managing networks and IP addresses. OpenStack Networking gives users self-service control over their network configurations. Users can define, separate, and join networks on demand. This allows for flexible network models that can be adapted to fit the requirements of different applications.

OpenStack Networking has a pluggable architecture that supports numerous virtual networking technologies as well as native Linux networking mechanisms including Open vSwitch and `linuxbridge`. OpenStack Networking is composed of several services. The `neutron-server` exposes the API and responds to user requests. The `neutron-l3-agent` provides L3 functionality, such as routing, through interaction with the other networking plugins and agents. The `neutron-dhcp-agent` provides DHCP to tenant networks. There are also a series of network agents that perform local networking configuration for the node's virtual machines.

This reference architecture is based on the Open vSwitch plugin, which uses the `neutron-openvswitch-agent`.

Horizon – Dashboard

The OpenStack Dashboard is an extensible web-based application that allows cloud administrators and users to control and provision compute, storage, and networking resources. Administrators can use the Dashboard to view the state of the cloud, create users, assign them to tenants, and set resource limits. The OpenStack Dashboard runs as an Apache web server via the `httpd` service.

Figure 4. OpenStack Dashboard

Services not covered in this reference architecture

Heat – Orchestration service

This service provides a REST API to orchestrate multiple composite cloud applications through a single template file. These templates allow for the creation of most OpenStack resource types such as virtual machine instances, floating IPs, volumes, and users. The Orchestration service is also tech preview in Red Hat Enterprise Linux OpenStack Platform 4 and not included in this reference architecture.

Swift – Object Storage service

The OpenStack Object Storage service provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. It provides redundant, scalable object storage using clusters of standardized servers capable of storing petabytes of data. Object Storage is not a traditional file system, but rather a distributed storage system for static data. Objects and files are written to multiple disks spread throughout the data center. Storage clusters scale horizontally simply by adding new servers. The OpenStack Object Storage service is not discussed in this reference architecture.

Supporting technologies

This section describes the supporting technologies used to develop this reference architecture beyond the OpenStack services and core operating system. Supporting technologies include:

MySQL

A state database resides at the heart of an OpenStack deployment. This SQL database stores most of the build-time and run-time state information for the cloud infrastructure including available instance types, networks, and the state of running instances in the compute fabric. Although OpenStack theoretically supports any SQL-Alchemy compliant database, Red Hat

Enterprise Linux OpenStack Platform 4 uses MySQL, a widely used open source database packaged with Red Hat Enterprise Linux 6.

Qpid

Enterprise messaging systems let programs communicate by exchanging messages. OpenStack services use enterprise messaging to communicate tasks and state changes between endpoints, schedulers, services and agents. Red Hat Enterprise Linux OpenStack Platform 4 uses Qpid for open source enterprise messaging. Qpid is an Advanced Message Queuing Protocol (AMQP) compliant, cross-platform enterprise messaging system developed for low latency based on an open standard for enterprise messaging. Qpid is released under the Apache open source license.

KVM

Kernel-based Virtual Machine (KVM) is a full virtualization solution for Linux on x86 and x86_64 hardware containing virtualization extensions for both Intel® and AMD processors. It consists of a loadable kernel module that provides the core virtualization infrastructure. Red Hat Enterprise Linux OpenStack Platform Compute uses KVM as its underlying hypervisor to launch and control virtual machine instances.

Packstack

Packstack is a Red Hat Enterprise Linux OpenStack Platform 4 installer. Packstack uses Puppet modules to install OpenStack packages via SSH. Puppet modules ensure OpenStack can be installed and expanded in a consistent and repeatable manner. This reference architecture uses Packstack for a multi-server deployment. Through the course of this reference architecture, the initial Packstack installation is modified with OpenStack Network and Storage service enhancements.

Open vSwitch

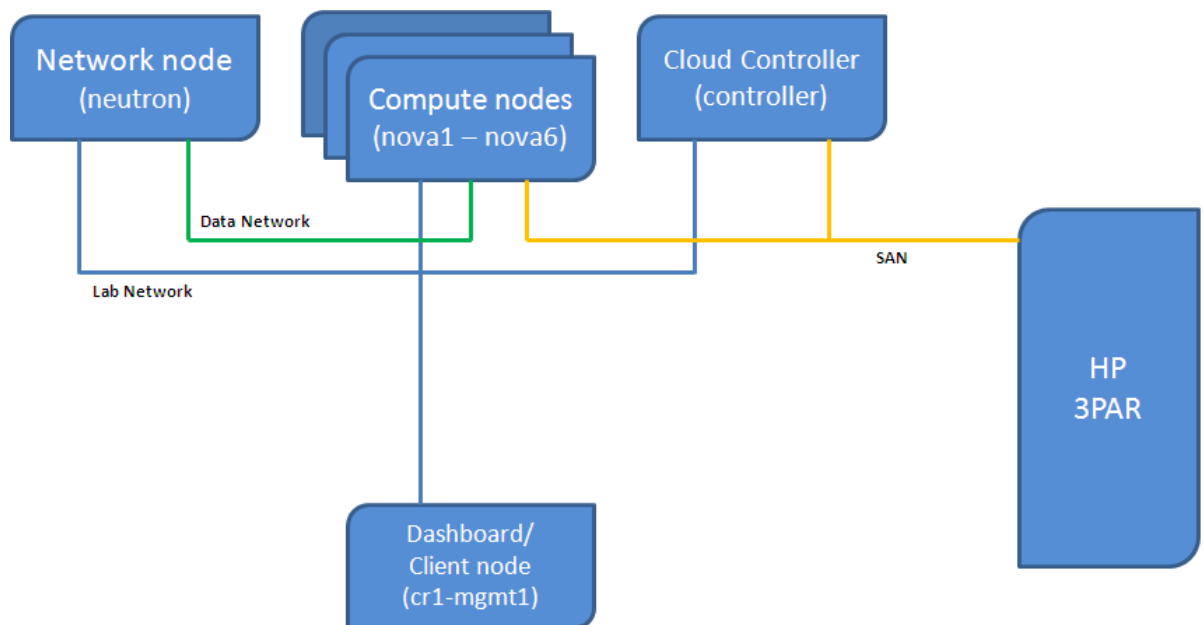
Open vSwitch is a production-quality, multilayer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols. In addition, it is designed to support distribution across multiple physical servers. Red Hat Enterprise Linux OpenStack Platform 4 provides an Open vSwitch plugin for Neutron that provides next-generation software networking infrastructure for both public and private clouds.

Deployment model

Network topology

Figure 5 shows the network topology used for this reference architecture.

Figure 5. Network topology



All servers are connected over the Lab Network switch – 10.64.80.0/20. This network is used for client requests to the API servers as well as service communication between the OpenStack services.

The network node and compute nodes are connected via a 10 GbE network on the Data network. This network carries the communication between virtual machines in the cloud and also carries all communications between the software-defined networking components. In this specific reference architecture, it is a switch configured to trunk a range of VLAN tags between the compute and network nodes.

The controller and compute nodes are connected to HP 3PAR via a storage area network. HP 3PAR provides the backend storage for the image service (glance) as well as persistent storage for the VMs via block storage service (cinder).

OpenStack Service placement

The table below shows the final service placement for all OpenStack services. The API-listener services (including quantum-server) run on the cloud controller in order to field client requests. The Network node runs all other Network services except for those necessary for Nova client operations, which also run on the Compute nodes.

Table 2. OpenStack final service placement

Component	Hostname	Role	Service
BL460c Gen8 (Blade 1)	controller	Cloud Controller	openstack-cinder-api openstack-cinder-scheduler openstack-cinder-volume openstack-glance-api openstack-glance-registry openstack-glance-scrubber openstack-keystone openstack-nova-api openstack-nova-cert openstack-nova-conductor openstack-nova-consoleauth openstack-nova-novncproxy openstack-nova-scheduler quantum-server
BL460c Gen8 (Blade 2)	neutron	Network node	neutron-dhcp-agent neutron-l3-agent neutron-lbaas-agent neutron-metadata-agent neutron-openvswitch-agent
BL460c Gen8 (Blades 3 – 8)	nova1 – nova6	Compute node	neutron-openvswitch-agent openstack-ceilometer-compute openstack-nova-compute
DL360p Gen8	cr1-mgmt1	Client/Dashboard	httpd openstack-ceilometer-alarm-evaluator openstack-ceilometer-alarm-notifier openstack-ceilometer-api openstack-ceilometer-central openstack-ceilometer-collector

Installation

HP hardware configuration

HP Integrated Lights-Out (iLO)

ProLiant servers provide exceptional remote management capabilities through the HP Integrated Lights-Out (iLO) solution. Make sure that you connect each system's iLO to your management network. Some key features that you may find helpful during OpenStack deployment include the Integrated Remote Console (IRC) and remote reset and power control. Console access via the integrated remote console (IRC) can be especially valuable during remote network configuration and troubleshooting. For more information about iLO configuration and features you can go to the general iLO web page at hp.com/go/iLO or visit the support page for your individual server.

Storage configuration for boot disk

All servers in this reference architecture are specified with multiple 300 GB physical drives. Each server is configured with an HP Smart Array controller, and we will use that to configure the available physical drives into a logical drive with your preferred RAID configuration. As shown in Figure 6, this logical drive will be used as a boot disk in this implementation.

Figure 6. Smart Array controller configuration

The screenshot shows the HP iLO 4 web interface for a ProLiant DL360p Gen8 server. The left sidebar contains a navigation menu with options like Information, iLO Federation, Remote Console, Virtual Media, Power Management, Network, Remote Support, and Administration. The main content area is titled 'System Information - Storage Information' and includes tabs for Summary, Fans, Temperatures, Power, Processors, Memory, Network, Storage, and Firmware. The 'Storage' tab is selected, showing 'Storage Information' with 'Logical View' selected. A note states: 'Note: This view shows configured logical drives and associated physical drives. It does not show physical drives which are not configured as part of an array, or spare drives.' Below this, there are expandable sections for 'Controller on System Board', 'Drive Enclosure Port 11 Box 1', 'Drive Enclosure Port 21 Box 1', and 'Logical Drive 01'. The 'Logical Drive 01' section shows the following details:

Controller Status	OK
Serial Number	001438024FB8A30
Model	HP Smart Array P420i Controller
Firmware Version	5.22
Cache Module Status	OK
Cache Module Serial Number	PBKUD0ARH3P2ZZ
Cache Module Memory	2097152 KB

Below the controller details, the 'Logical Drive 01' section shows:

Status	OK
Capacity	1117 GB
Fault Tolerance	RAID 50
Encryption Status	Not Encrypted

This configuration provides good I/O performance and data protection for the server boot drive, database, message queue and services on the controller. For the Compute services the RAID 50 configuration will be a benefit because we are using local storage as boot disk with nova services.

Storage connection to blades

Controller and compute nodes need block storage access. The glance service running on the controller node needs storage space to store images. An HP 3PAR volume must be created and presented to the controller node. Compute nodes which run VM instances must have a path to HP 3PAR for VMs to access persistent storage.

Virtual Connect Manager is used to configure SAN Fabrics that define storage connections from server blades to HP 3PAR, as shown in Figure 7.

Figure 7. Virtual Connect SAN Fabric

Edit SAN Fabric: CR1_JC1

Fabric

Name	Status	Type	Login Re-Distribution	Configured Speed	☐ Show Advanced Settings
CR1_JC1	✔	FabricAttach	MANUAL	4 Gb	

Enclosure Uplink Ports

Uplink Port	Enclosure	Bay	Port Status	Connected To	Action
Port X1	CR1_Enc1	1	✔ LOGGED-IN	4 Gb 10:00:00:27:f8:6d:d9:fa	Delete
Port X2	CR1_Enc1	1	✔ LOGGED-IN	4 Gb 10:00:00:27:f8:6d:d9:fa	Delete
Port X3	CR1_Enc1	1	✔ LOGGED-IN	4 Gb 10:00:00:27:f8:6d:d9:fa	Delete

Add Port

Bay 1 »

Apply

Cancel

Network configuration for server blades

Use the Virtual Connect Manager to configure network connections on server blades. Set up network connections as per the network topology design described earlier. The first step is to configure a shared uplink. These uplinks connect to the Lab Network via 10 GbE switches (ToR). Define a shared uplink as shown in Figure 8.

Figure 8. Virtual Connect Shared Uplink Set

Edit Shared Uplink Set: CR1_E1_IC1_SUS

Ethernet Shared External Uplink Set

Uplink Set Name	Status	PID
CR1_E1_IC1_SUS		

External Uplink Ports

Port	Port Role	Port Status	Connector Type	Connected To	PID	Speed/Duplex	Action
CR1_Enc1 Bay 1: Port X5	NA	Linked-Active	10 Gb SFP-DAC	Unknown ()		Auto	Delete
CR1_Enc1 Bay 1: Port X6	NA	Linked-Active	10 Gb SFP-DAC	Unknown ()		Auto	Delete

Connection Mode: ☒ Auto ☐ Failover
 LACP Timer: ☒ Domain Default, Short (1 sec) ☐ Short (1 sec) ☐ Long (30 sec)

Add Port

CR1_Enc1

Associated FCoE Network (VLAN tagged)

+ Add
Edit
Delete

Associated Networks (VLAN tagged)

+ Add
Delete

☐	Network Name	VLAN ID	Native	Smart Link	Private Network	Action
☐	CR1_E1_IC1_Data	120	true	true	false	Edit
☐	CR1_E1_IC1_DC_Lab	64	false	true	false	Edit
☐	ovs_vlan1000	1000	false	false	false	Edit
☐	ovs_vlan1001	1001	false	false	false	Edit

Table 3 describes the VLANs used for this reference architecture. Define the following VLANs listed in Table 3 using the +Add button on the Associated Networks (VLAN tagged) section as shown in Figure 9.

Table 3. VLANs used in reference architecture for Network Topology

Network	Name	VLAN	Purpose
Lab	CR1_E1_IC1_DC_Lab	64	Lab network for communication between servers and OpenStack services
Data	CR1_E1_IC1_Data	120	Communication between OpenStack Networking components in Compute and Network node and all VM traffic.
Tenants	ovs_vlan10xx	1000-1050	Data network for tenants. Define VLAN for every OpenStack tenant.

Figure 9. Create Associated Networks

Associated Networks (VLAN tagged)

Would you like to add...

☒ a single Associated Network
 ☐ multiple Associated Networks

Network Name *

ovs_vlan_1002

VLAN ID *

1002

Color

red

Labels

Type to add Network Labels

Type to add Network Labels

☐ Native
 ☐ Smart Link
 ☐ Private Network

☐ Advanced Network Settings

Type network access group names

Default

Type letters or numbers ('a', 'z', 'e', 'nag', 'default')

Apply

Cancel

Next, configure the blade servers to make use of the defined Ethernet and SAN fabric connections. Using Virtual Connect Manager, define a Server profile as shown in Figure 10. Specify the Lab, Data and Tenant network under the Ethernet Adapter Connections. For SAN connections, specify SAN fabric under FCoE HBA Connections. Create server profiles for all blade servers. Do not define SAN fabrics for the blade hosting the network (neutron) services.

Figure 10. Virtual Connect Server Profile

Edit Server Profile: Compute_01

Profile

Profile Name	Network Access Group	Status	Serial Number	Server UUID
Compute_01	Default		VCX0000V00	eb19dd73-9e2e-427c-89bc-761f63eb5700

☐ Hide Unused FlexNICs

Ethernet Adapter Connections

Port	Network Name	Status	Port Speed Type	Allocated Port Sp...	PXE	Multicast Filter	MAC	Mapping	Action
1	CR1_E1_IC1_DC_Lab		PREFERRED	6 Gb - 10 Gb	USE-BIOS	None	00-17-A4-77-7...	LOM1:1-a => Bay 1:d1:...	
2	Multiple Networks...		CUSTOM...	1 Gb - 10 Gb	USE-BIOS	None	00-17-A4-77-7...	LOM1:2-a => Bay 2:d1:...	

+ Add

iSCSI HBA Connections

Port	Network Name	Status	Port Speed Type	Allocated Port Speed ...	Boot Setting	MAC	Mapping	Action
------	--------------	--------	-----------------	--------------------------	--------------	-----	---------	--------

+ Add

FCoE HBA Connections

Port	Connects	FC SAN / FCoE Network Name	Type	Status	Port Speed Type	Allocated...	VW/PN	MAC	Mapping	Action
1	Bay 1	CR1_IC1	SAN		PREFERRED	4 Gb - 8 G	50:06:0B:00:00:C2:DE:00	00-17-A4-77-7C-0C	LOM1:...	
2	Bay 2	CR1_IC2	SAN		PREFERRED	4 Gb - 8 G	50:06:0B:00:00:C2:DE:02	00-17-A4-77-7C-08	LOM1:...	Delete

+ Add
☐ Fibre Channel Boot Parameters

Assign Profile to Server Bay

Enclosure	Server	Model	SN	Status	Power	UID
CR1_Enc1	Bay 1	ProLiant BL460c Gen8	USE309X6N8			

While defining Ethernet connections in a server profile, configure Multiple Networks for the second Ethernet connection. This connection must be updated for every new tenant VLAN you create. Ensure you create enough VLANs and add them under the Multiple Networks as shown in Figure 11.

Figure 11. Edit Multiple Networks

Server VLAN Tag to vNet Mappings ?

☐ Force the same VLAN mappings as in the Shared Uplink Set None

Networks not in mapping

Alphabetical

Filtered

AllA-CD-FG-IJ-LM-OP-RS-VW-Z0-34-67-9

CR1_E1_IC1_DC_Lab

CR1_E1_IC2_DC

CR1_E1_IC2_Mgmt

Data_Network

✓

✓

✓

List

Detail

Networks in mapping (3 mapped; limit is 162)

Drag and drop networks to include them or remove them from the mapping.

vNet Name	Status	Uplink VLAN Id	Server VLAN Id	Untagged
CR1_E1_IC1_Data	✓	120	120	☐
ovs_vlan1000	✓	1000	1000	☐
ovs_vlan1001	✓	1001	1001	☐
Drag and Drop networks				

OK

Cancel

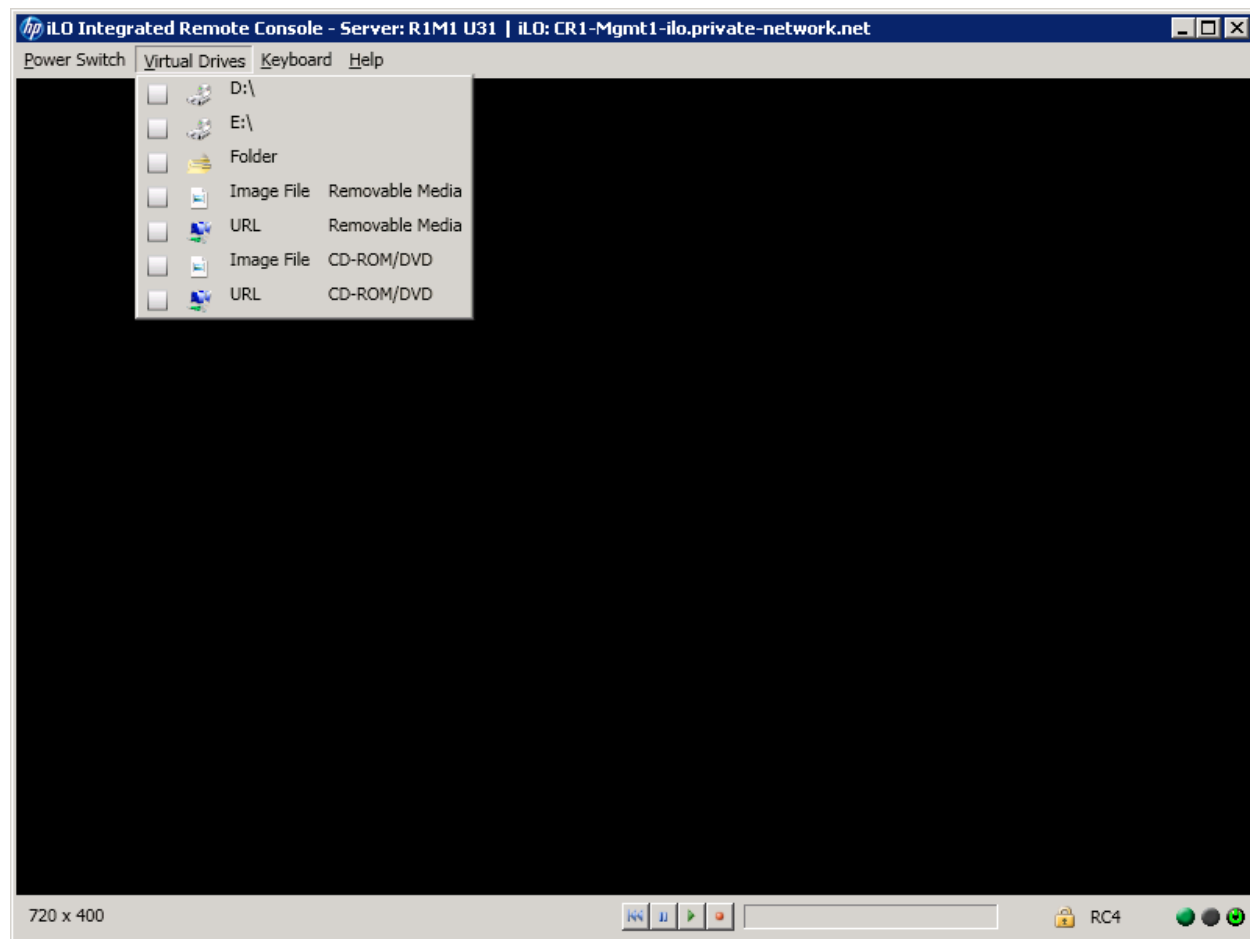
Network configuration for DL360p Gen8

Set up the DL360p Gen8 with one Ethernet port and connect this port to the Lab Network.

Operating system deployment and configuration

Install Red Hat Enterprise Linux 6.5 using the iLO with a DVD media. Open the Remote Console from the iLO and configure a Virtual Drive Image File CD-ROM/DVD option to mount the installation media. Boot the server from the installation media and complete the installation.

Figure 12. Mount Image File in iLO



Note

Other methods of installation, such as using a PXE server, can also be employed. Ensure a consistent installation on all servers.

After Red Hat Enterprise Linux 6.5 installation is complete, configure hostnames and NICs on servers as shown in Table 4. Configure /etc/hosts or DNS to reflect these settings.

Table 4. Host names and IP addresses

Hostname	Role	Network/Interface	IP address
controller	Cloud controller (Block storage)	Lab/eth0	10.64.80.83
		Data/eth1	10.64.80.83
neutron	Network	Lab/eth0	10.64.80.84
		Data/eth1	VLANs 1000-1050
nova1	Compute	Lab/eth0	10.64.80.85
		Data/eth1	VLANs 1000-1050
nova2	Compute	Lab/eth0	10.64.80.86
		Data/eth1	VLANs 1000-1050
nova3	Compute	Lab/eth0	10.64.80.87
		Data/eth1	VLANs 1000-1050
nova4	Compute	Lab/eth0	10.64.80.88
		Data/eth1	VLANs 1000 - 1050
nova5	Compute	Lab/eth0	10.64.80.89
		Data/eth1	VLANs 1000-1050
nova6	Compute	Lab/eth0	10.64.80.90
		Data/eth1	VLANs 1000-1050
Cr1-mgmt1	Dashboard/client	Lab/eth0	10.64.80.81
HP 3PAR		Lab	10.64.80.237

Note

Be sure to enable the corresponding VLAN IDs on all Ethernet switches as necessary. If not, connections to the servers or the VM instances deployed using OpenStack will not be available.

Configure the eth0 interface on all nodes to start on boot and use a static IP. The interface configuration file /etc/sysconfig/network-scripts/ifcfg-eth0 for controller node is as shown below.

```
DEVICE=eth0
HWADDR=00:17:A4:77:7C:00
TYPE=Ethernet
ONBOOT=yes
NM_CONTROLLED=no
BOOTPROTO=static
IPADDR=10.64.80.83
NETMASK=255.255.240.0
GATEWAY=10.64.80.1
```

Specifically on the network node, configure a bridge interface br-ex, which will be used by OpenStack as external network. The br-ex interface is defined in file /etc/sysconfig/network-scripts/ifcfg-br-ex as shown below.

```
DEVICE=br-ex
DEVICETYPE=ovs
TYPE=OVSBridge
NM_CONTROLLED=no
BOOTPROTO=static
IPADDR=10.64.80.84
NETMASK=255.255.240.0
GATEWAY=10.64.80.1
```

The eth0 interface on the network node must be defined as an Open vSwitch port as shown below in the file /etc/sysconfig/network-scripts/ifcfg-eth0.

```
DEVICE=eth0
ONBOOT=yes
TYPE=OVSPort
DEVICETYPE=ovs
NM_CONTROLLED=no
BOOTPROTO=none
OVS_BRIDGE=br-ex
```

Restart networking:

```
$ service network restart
```

Note

A provider network can also be used instead of the above shown bridge configuration. A provider network maps directly to a physical network in the data center. They are used to give tenants direct access to public networks.

Configure software repositories

Once the network is set up, register all servers to Red Hat Network and add the necessary subscriptions. Table 5 details the mandatory channels that must be subscribed.

Table 5. Mandatory subscription channels

Channel	Repository Name
Red Hat OpenStack 4.0 (RPMs)	rhel-6-server-openstack-4.0-rpms
Red Hat Enterprise Linux 6 Server (RPMs)	rhel-6-server-rpms

You can now verify if the above channels are subscribed by analyzing the output of the “yum repolist” command. Table 6 lists the repos that must be in the output of the command.

Table 6. Repositories for command output

Repo ID	Repository Name
rhel-6-server- openstack-Red Hat OpenStack 4.0 (RPMs)	Red Hat OpenStack 4.0 (RPMs)
rhel-6-server-rpms	Red Hat Enterprise Linux 6 Server (RPMs)

For more details on how to add channels and subscriptions refer to section 2.1.2 in the Red Hat Enterprise Linux OpenStack Platform 4 – Getting Started Guide.

Finally, update all servers.

```
$ yum -y update
```

Configure multipath

Install and configure multipath on all servers that need connection to storage on HP 3PAR. Use the sample configuration below, /etc/multipath.conf, as a reference.

```
devices {
    device {
        vendor "3PARdata"
        product "VV"
        no_path_retry 18
        features "0"
        hardware_handler "0"
        path_grouping_policy multibus
        getuid_callout "/lib/udev/scsi_id --whitelisted --device=/dev/%n"
```

```

        path_selector "round-robin 0"
        rr_weight uniform
        rr_min_io_rq 1
        path_checker tur
        failback immediate
    }
}

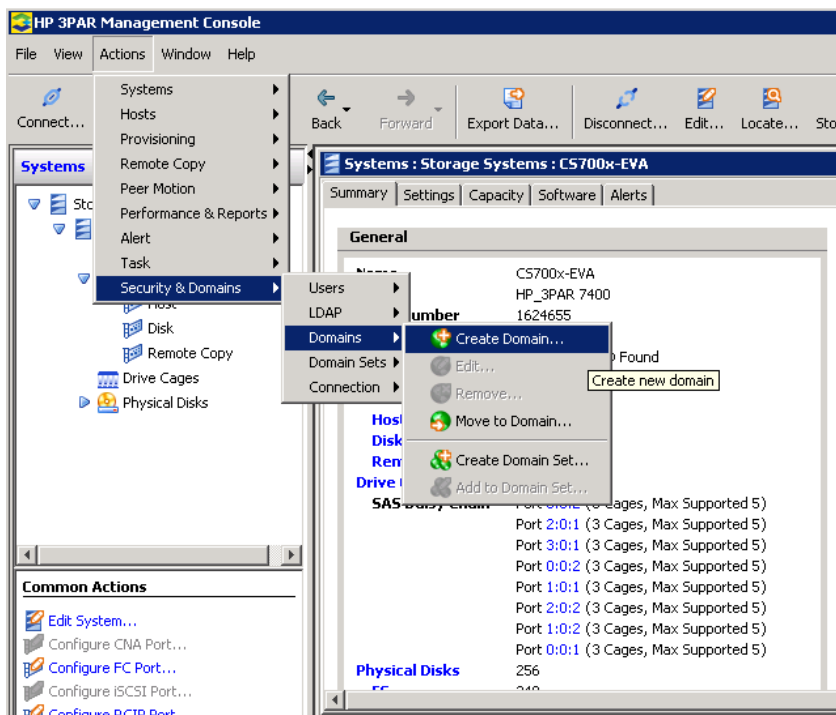
```

Enable and restart the multipathd service after the configuration is applied to the controller and compute nodes. Reboot nodes as necessary.

Configure HP 3PAR

Create a Domain "rhos_d0" on HP 3PAR to host all volumes that are created for use by the Red Hat OpenStack services. Launch HP 3PAR Management Console installed on the jumpstation. Navigate to Actions → Security & Domains → Domains → Create Domain. This will pop-up a window to create the domain.

Figure 13. HP 3PAR domain creation



On this window, specify the domain name and any comments optionally. Click on the Add button below the comments input box. This will add the domain to the list of new domains. Click OK to confirm and add a new domain.

Figure 14. Create Domain

Create Domain

Domain Properties

Storage Systems: CS700x-EVA (1624655)

Domain Name:

☒ Maximum Volume Retention Time (1-1,825) ☐ Override System 14 day(s)

Comments:

Add Remove

New Domains

Domain	System	Maximum Volume Retention Time	Comments
rhos_d01	CS700x-EVA		-- Domain for Red Hat OpenStack

Help OK Cancel

Next, create a 3PAR common provisioning group (CPG) under the newly created domain and name it `cpg_rhos`.

Figure 15. Create CPG

HP 3PAR Management Console

File View Actions Window Help

Connect... Refresh Maximize Home Back Forward Export Data... Create Edit... Remove... Save As Template... Compact... Move to Domain...

Provisioning

- Storage Systems
- Domains
 - rhos_d0
 - CPGs
 - Virtual Volumes
 - Remote Copy
 - Exported
 - Unexported
 - By Device Type
 - By RAID Type
 - Virtual Volume Sets
 - VLLUNs
 - AO Configurations

Common Actions

- Create Virtual Volume...
- Create Virtual Copy...
- Create Physical Copy...
- Create CPG...

Provisioning : Domains : rhos_d0 : CPGs

Summary CPGs Alerts

Summary 1 object Filter Clear Any column

Name	System	State	RAID	Allocated Capacity (GiB)	Used Capacity (GiB)	Growth Warning (GiB)	Growth Limit (GiB)	Total VV Count
cpg_rhos	CS700x-EVA	Normal	RAID 5	484.000	402.250	<Disabled>	<Disabled>	2

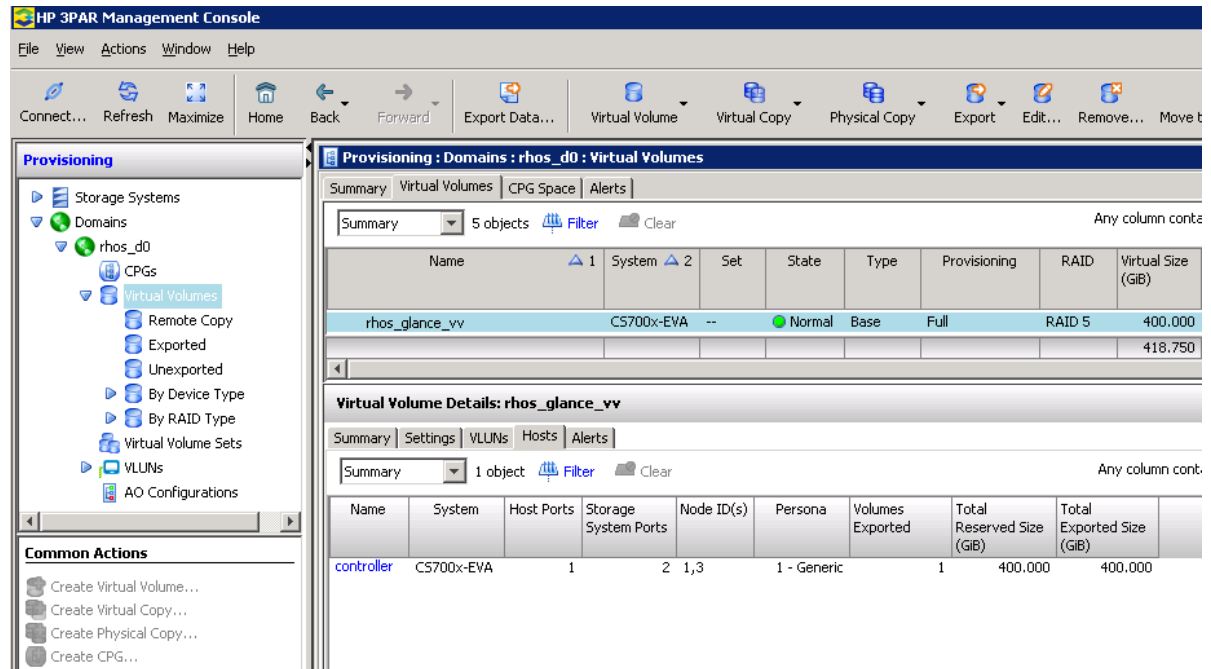
CPG Details: cpg_rhos

Summary Settings Virtual Volumes Alerts

General		Capacity	
Name	cpg_rhos	Device Type	
Domain	rhos_d0	Overview	Allocated
Total VV Count	2		
Fully Provisioned	1		
Thinly Provisioned	1		

Create a virtual volume under the rhos_d0 domain and present it to the cloud controller server. It is on this controller server that glance services run and are configured to store all images on this newly created virtual volume.

Figure 16. Create Virtual Volume



Please reference the Red Hat Release Notes at:

https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/4/html-single/Release_Notes/index.html

Red Hat OpenStack installation and configuration

Install Packstack

Packstack is a command-line utility that uses Puppet modules to enable rapid deployment of OpenStack on existing servers over an SSH connection. Deployment options are provided either interactively, via the command line, or non-interactively by means of a text file containing a set of preconfigured values for OpenStack parameters.

Packstack is suitable for deploying the following types of configurations:

- Single-node proof-of-concept installations, where all controller services and your virtual machines run on a single physical host. This is referred to as an all-in-one install.
- Proof-of-concept installations where there is a single controller node and multiple compute nodes. This is similar to the all-in-one install above, except you may use one or more additional hardware nodes for running virtual machines.

Packstack is provided by the openstack-packstack package. Follow this procedure to install the openstack-packstack package on the client server.

1. Use yum command to install Packstack

```
$ yum install openstack-packstack
```

2. Verify Packstack is installed

```
$ which packstack
```

```
/usr/bin/packstack
```

Running Packstack deployment utility

The steps below outline the procedure to run Packstack. Run the following commands on the client server.

1. Generate packstack answer file.

```
$ packstack --gen-answer-file=packstack.txt
```

2. Edit packstack answer file to key in the values. Refer to Appendix A for the values that were used for this reference architecture.

```
$ vi packstack.txt
```

3. Run the packstack utility providing the answer file as input.

```
$ packstack --answer-file=packstack.txt
```

4. After the run is complete, you should see a success message and no errors displayed. This may take a few minutes depending on the number of compute servers to be configured. Observe the progress on the console.

```
**** Installation completed successfully ****
```

5. Reboot all servers.
6. Packstack creates a demo tenant and configures a password as provided in the answer file.
7. When the servers come back up, log into the Horizon dashboard on the client server using user demo to verify the installation, <http://10.64.80.81/dashboard>
8. Packstack creates a `keystonerc_admin` file for admin user in the home directory of the node where packstack is run. Create a new identity for demo user by copying the `keystonerc_admin` file to `keystonerc_demo`. Edit the file to change user from admin to demo, change the password as appropriate. These files are sourced when running OpenStack commands for authentication purposes.

Key point

You can as well run Packstack interactively and provide input on the command line. Use the answer file as a reference and key-in input accordingly.

Configure Glance

Configure Glance to use a virtual volume that was created earlier on HP 3PAR. In this reference architecture glance service is hosted on the controller node.

1. Configure a filesystem on the new disk on the controller node.

```
$ mkfs.ext4 /dev/mapper/mpathb
```

2. Glance places all images under `/var/lib/glance/images`. Mount the new disk on path `/var/lib/glance/images`

```
$ mount /dev/mapper/mpathb /var/lib/glance/images
```

3. Log in to <https://rhn.redhat.com/rhn/software/channel/downloads/Download.do?cid=16952> with your Customer Portal user name and password and download the KVM Guest Image

4. Switch to demo identity

```
$ source keystonerc_demo
```

5. Upload the image file. Below is a command to upload the image.

```
$ glance image-create --name "RHEL65" --is-public true --disk-format qcow2 \
--container-format bare --file rhel-guest-image-6.5-20140307.0.x86_64.qcow2
```

Note

You can use the dashboard UI to upload the image. Log in as admin or demo user and upload the downloaded image. Add any additional images that you may need for testing, for example, CirrOS 0.3.1 image in qcow2 format.

Configure Cinder and HP3PARFCDriver

The HP3PARFCDriver gets installed with the OpenStack software on the controller node.

1. Install the `hp3parclient` Python package on the controller node. Either use `pip` or `easy_install`. This version of Red Hat OpenStack, which is based on Havana, requires version 2.0.

```
$ pip install hp3parclient==2.0
```

2. Verify that the HP 3PAR Web Services API server is enabled and running on the HP 3PAR storage system. Log onto the HP 3PAR storage system with administrator access.

```
$ ssh 3paradm@10.64.80.237
```

3. View the current state of the Web Services API Server.

```
$ showwsapi
-Service-  -State-  -HTTP_State-  HTTP_Port  -HTTPS_State-  HTTPS_Port  -
Version-
Enabled    Active    Enabled      8008        Enabled      8080
1.1
```

If the Web Services API Server is disabled, start it.

```
$ startwsapi
```

If the HTTP or HTTPS state is disabled, enable one of them.

```
$ setwsapi -http enable
```

or

```
$ setwsapi -https enable
```

4. If you are not using an existing CPG, create a CPG on the HP 3PAR storage system to be used as the default location for creating volumes.
5. On the controller node where the cinder service is run, edit the `/etc/cinder/cinder.conf` file and add the following lines. This configures HP 3PAR as a backend for persistent block storage. Ensure to configure the right HP 3PAR username and password.

```
[3parfc]
volume_driver=cinder.volume.drivers.san.hp.hp_3par_fc.HP3PARFCDriver
volume_backend_name=3par_FC
hp3par_api_url=https://10.64.80.237:8080/api/v1
hp3par_username=<<3par username>>
hp3par_password=<<3par user password>>
hp3par_cpg=cpg_rhos
san_ip=10.64.80.237
san_login=<<3par username>>
san_password=<<3par user password>>
```

6. Restart the cinder volume service.

```
$ service openstack-cinder-volume restart
```

Note

For more details on HP 3PAR StoreServ block storage drivers and to configure multiple HP 3PAR storage backends refer to the “OpenStack® HP 3PAR StoreServ Block Storage Drivers Configuration Best Practices” document available at <http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA4-9853ENW>. More advanced configuration with “Volume Types” is available in the guide on creating OpenStack cinder type-keys.

The HP3PARFCDriver is based on the Block Storage (Cinder) plug-in architecture. The driver executes the volume operations by communicating with the HP 3PAR storage system over HTTP/HTTPS and SSH connections. The HTTP/HTTPS communications use the hp3parclient, which is part of the Python standard library.

Configure security group rules

Security groups control access to VM instances. Define protocol level access to VM instances using Security Groups. Navigate to Manage Compute → Access & Security → Security Groups. Edit the default security group. Click on the +Add Rule button to add new rules into the default security group as shown below. Ensure SSH and ICMP protocols are configured to allow traffic from the public and private network.

Note

For more details on HP 3PAR StoreServ block storage drivers and to configure multiple HP 3PAR storage backends refer to the “OpenStack® HP 3PAR StoreServ Block Storage Drivers Configuration Best Practices” document available at <http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA4-9853ENW>. More advanced configuration with “Volume Types” is available in the guide on creating OpenStack cinder type-keys.

Figure 17. Add Rule

Add Rule

Rule *

SSH

Remote *

CIDR

CIDR

192.168.32.0/24

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

Cancel

Add

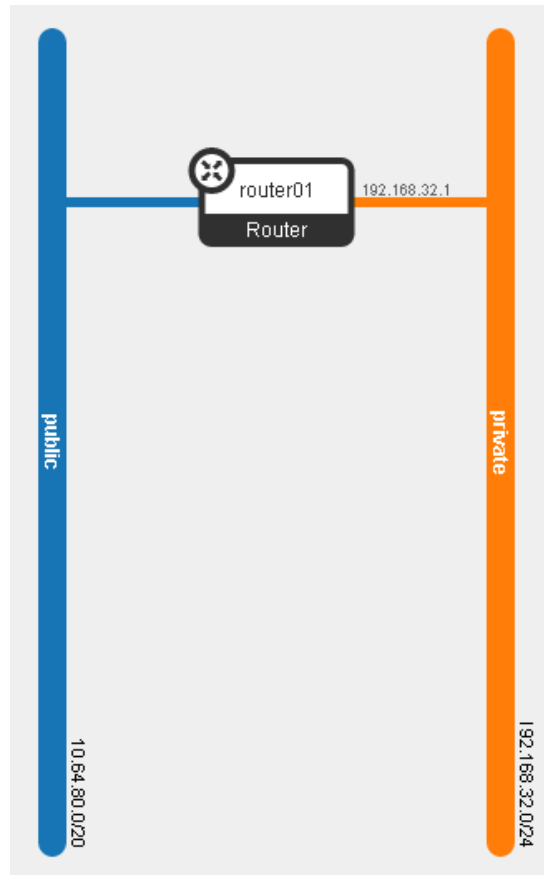
Note

For troubleshooting purposes add Custom TCP Rules for both Ingress and Egress directions allowing port range 1 – 65535 to CIDR 0.0.0.0/0.

Configure OpenStack networking

VM instances deployed on the compute nodes make use of the host neutron as network server. All VM traffic from compute nodes use the neutron server for communication. The neutron server does all the switching and routing between the VMs as well as route between external clients and the VM instances. OpenStack networking configuration in this reference architecture makes use of two networks (private and public), two subnets (public_sub and priv_sub) and a virtual router (router01). Post configuration, the network configuration will be as shown in Figure 18. The `private/priv_sub` network is defined to be a network for internal and VM traffic. For external communication the `public/public_sub` network will be used.

Figure 18. OpenStack network topology



During the Packstack installation all necessary Open vSwitch configurations will be created on the neutron server. Ensure the following entries are already configured under the OVS section in the `/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini` file.

```
[OVS]
vxlan_udp_port=4789
network_vlan_ranges=physnet1:1000:1050
tenant_network_type=vlan
enable_tunneling=False
integration_bridge=br-int
bridge_mappings=physnet1:br-eth1
```

Run the command below to ensure `eth0` exists as a port under bridge `br-ex`.

```
[root@neutron ~]# ovs-vsctl show
00c91a3f-47a5-439a-b27a-648db5b1e7c0
Bridge "br-eth1"
Port "eth1"
Interface "eth1"
Port "phy-br-eth1"
Interface "phy-br-eth1"
Port "br-eth1"
```

```

Interface "br-eth1"
type: internal
Bridge br-int
Port br-int
Interface br-int
type: internal
Port "int-br-eth1"
Interface "int-br-eth1"
Bridge br-ex
Port br-ex
Interface br-ex
type: internal
Port "eth0"
Interface "eth0"
ovs_version: "1.11.0"

```

At this point, we are ready to create OpenStack networking elements. The steps below list all commands to run to create public and private networks, create public_sub and priv_sub subnets, create a virtual router, and create routing between private and public networks.

1. Switch to admin identity:

```
[root@neutron ~]# source keystone_admin
```

2. Create a public network:

```
[root@neutron ~(keystone_admin)]# neutron net-create public --shared --
router:external=True
```

3. Create a subnet under public network:

```
[root@neutron ~(keystone_admin)]# neutron subnet-create --name public_sub --
enable-dhcp=False --allocation-pool start=10.64.80.200,end=10.64.80.250 --
gateway=10.64.80.1 public 10.64.80.0/20
```

4. Switch to demo identity:

```
[root@neutron ~(keystone_admin)]# source keystone_demo
```

5. Create a private network:

```
[root@neutron ~(keystone_demo)]# neutron net-create private
```

6. Create a subnet under private network for VM traffic:

```
[root@neutron ~(keystone_demo)]# neutron subnet-create --name priv_sub --
enable-dhcp=True private 192.168.32.0/24
```

7. Create a virtual router:

```
[root@neutron ~(keystone_demo)]# neutron router-create router01
```

8. Add the private subnet to the router:

```
[root@neutron ~(keystone_demo)]# neutron router-interface-add router01
priv_sub
```

9. Switch back to admin identity:

```
[root@neutron ~(keystone_demo)]# source keystone_admin
```

10. Set the public network as gateway to the router:

```
[root@neutron ~(keystone_admin)]# neutron router-gateway-set router01 public
```

Verify private network connectivity

1. Ping the router's external interface – Run the following commands to determine if the router's external IP is reachable from the client server. Note that these commands make use of environment variables to store values to be used in subsequent commands.

- a. Determine router ID:

```
[root@CR1-Mgmt1 ~(keystone_demo)]# router_id=$(neutron router-list | awk
'/router01/ {print $2}')
```

- b. Determine private subnet ID:

```
[root@CR1-Mgmt1 ~(keystone_demo)]# subnet_id=$(neutron subnet-list | awk
'/192.168.32.0/ {print $2}')
```

- c. Determine router IP:

```
[root@CR1-Mgmt1 ~(keystone_demo)]# router_ip=$(neutron subnet-show
$subnet_id | awk '/gateway_ip/ {print $4}')
```

- d. Determine router network namespace on the neutron server. In this reference architecture, the network server is the neutron server.

```
[root@CR1-Mgmt1 ~(keystone_demo)]# qroute_id=$(ssh neutron ip netns list |
grep qrouter)
```

- e. Ping the external interface of the router within the network namespace on the network node. This proves network connectivity between the server and the router.

```
[root@CR1-Mgmt1 ~(keystone_demo)]# ssh neutron ip netns exec $qroute_id ping
-c 2 $router_ip
```

```
PING 192.168.32.1 (192.168.32.1) 56(84) bytes of data.
64 bytes from 192.168.32.1: icmp_seq=1 ttl=64 time=0.065 ms
64 bytes from 192.168.32.1: icmp_seq=2 ttl=64 time=0.034 ms
```

```
--- 192.168.32.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.034/0.049/0.065/0.017 ms
```

Validation**Launch an instance**

At this point, the OpenStack cloud is deployed and should be functioning. Point your browser to the public address of the OpenStack-dashboard node, "<http://10.64.80.81/horizon>", login as user demo.

As a first step, create a public keypair for SSH access to the instances. Navigate to Manage Compute → Access & Security → Keypairs → Click on the + Create Keypair button. Key in the keypair name as demokey. Download this keypair file and copy it to the client server from which instances can be accessed.

Figure 19. Creation of SSH Keypair

Create Keypair
×

Keypair Name *

Description:

Keypairs are ssh credentials which are injected into images when they are launched. Creating a new key pair registers the public key and downloads the private key (a .pem file).

Protect and use the key as you would any normal ssh private key.

Next, navigate to Manage Compute → Instances → Click on the + Launch Instance button. This will pop-up a window as shown below. Click on the Launch button to create an instance for the RHEL 6.5 image that was uploaded earlier.

Figure 20. Launch instance – Details tab

Launch Instance ×

Details *

Access & Security *

Networking *

Post-Creation

Availability Zone

nova ▼

Instance Name *

rhelvm1

Flavor *

m1.small ▼

Instance Count *

1

Instance Boot Source *

Boot from image ▼

Image Name

RHEL65 (337.7 MB) ▼

Specify the details for launching an instance.

The chart below shows the resources used by this project in relation to the project's quotas.

Flavor Details	
Name	m1.small
VCPUs	1
Root Disk	20 GB
Ephemeral Disk	0 GB
Total Disk	20 GB
RAM	2,048 MB

Project Limits

Number of Instances 3 of 10 Used

Number of VCPUs 3 of 20 Used

Total RAM 3,072 of 51,200 MB Used

Cancel

Launch

Under the Access & Security tab, select the demokey and check the default security group.

Figure 21. Launch instance – Access and Security tab

The screenshot shows the 'Launch Instance' dialog with the 'Access & Security' tab selected. The 'Keypair' dropdown is set to 'demokey'. The 'Admin Pass' and 'Confirm Admin Pass' fields are empty. The 'Security Groups' section shows the 'default' group checked. A description on the right states: 'Control access to your instance via keypairs, security groups, and other mechanisms.' At the bottom right are 'Cancel' and 'Launch' buttons.

Launch Instance ×

Details * Access & Security * Networking * Post-Creation

Keypair
demokey +

Admin Pass

Confirm Admin Pass

Security Groups *
☒ default

Control access to your instance via keypairs, security groups, and other mechanisms.

Cancel Launch

Under the Networking tab, configure to use private network by selecting and dragging up the “private” network name.

Figure 22. Launch instance – Networking

The screenshot shows the 'Launch Instance' dialog with the 'Networking' tab selected. The 'Selected Networks' section shows 'nic:1' connected to the 'private' network. The 'Available networks' section shows the 'public' network. A description on the right states: 'Choose network from Available networks to Selected Networks by push button or drag and drop, you may change nic order by drag and drop as well.' At the bottom right are 'Cancel' and 'Launch' buttons.

Launch Instance ×

Details * Access & Security * Networking * Post-Creation

Selected Networks

nic:1 ⇄ private (5b5541d2-333e-4e2a-5b2d-3624b6328665) -

Available networks

⇄ public (571b262-14b-e-4595a339-5c3ed-4764325) +

Choose network from Available networks to Selected Networks by push button or drag and drop, you may change nic order by drag and drop as well.

Cancel Launch

Once the instance is launched, the power state will be set to running if there were no errors during instance creation. Wait for a while for the VM instance to boot completely. Click on the instance name “rhelvm1” to view more details. On the same page navigate to the Console tab to view the VM instance console.

Figure 23. Instance status

Instances Logged in as: demo Settings Help Sign Out										
Instances										
Filter				Filter	+ Launch Instance		Soft Reboot Instances		Terminate Instances	
☐	Instance Name	Image Name	IP Address	Size	Keypair	Status	Task	Power State	Uptime	Actions
☐	rhelvm1	RHEL65	192.168.32.5	m1.small 2GB RAM 1 VCPU 20.0GB Disk	demokey	Active	None	Running	23 hours, 6 minutes	Create Snapshot More ▾

Verify routing

Follow the steps below to test network connectivity to the newly created instance from the client server on which you have copied the demokey keypair.

1. Determine the gateway IP of the router using the command below. The IP 10.64.80.200 is the gateway IP.

```
[root@CR1-Mgmt1 ~(keystone_demo)]# ssh neutron 'ip netns exec $(ip netns | grep
qrouter) ip a | grep 10.64.80'
inet 10.64.80.200/20 brd 10.64.95.255 scope global qg-e0836894-7e
```

2. Add a route to the private network on the public network via router's interface:

```
[root@CR1-Mgmt1 ~(keystone_demo)]# route add -net 192.168.32.0 netmask
255.255.255.0 gateway 10.64.80.200
```

3. SSH directly to the instance using private IP:

```
[root@CR1-Mgmt1 ~]# ssh -i demokey.pem cloud-user@192.168.32.5 uptime
The authenticity of host '192.168.32.5 (192.168.32.5)' can't be established.
RSA key fingerprint is cb:fe:eb:f8:67:18:f6:08:07:10:6e:e6:16:db:02:a4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.32.5' (RSA) to the list of known hosts.
04:23:12 up 23:12, 0 users, load average: 0.00, 0.00, 0.00
```

Add externally accessible IP

Add a floating IP from the public network to the newly created instance. For this you need to first create a floating IP. Navigate to Manage Compute → Access & Security → Floating IPs → Click on Allocate IP to Project. On the window that pops-up, select the public pool and click on Allocate IP.

Figure 24. Add a floating IP

Allocate Floating IP [Close]

Pool *
public ▼

Description:
Allocate a floating IP from a given floating IP pool.

Project Quotas
Floating IP (3) 47 Available

Cancel Allocate IP

On the same window, you will now see the newly created floating IP. Click on the Associate button under the Actions column. Select the rhelvm1 Port from the dropdown list and click on Associate.

Figure 25. Map floating IP

Manage Floating IP Associations [Close]

IP Address *

IP Address *
10.64.80.203 ▼ +

Port to be associated *
rhelvm1: 192.168.32.5 ▼

Select the IP address you wish to associate with the selected instance.

Cancel Associate

The Instances page will now show the floating IP associated with the rhelvm1 instance.

Figure 26. Instance status with floating IP

Instances Logged in as: demo

Instances

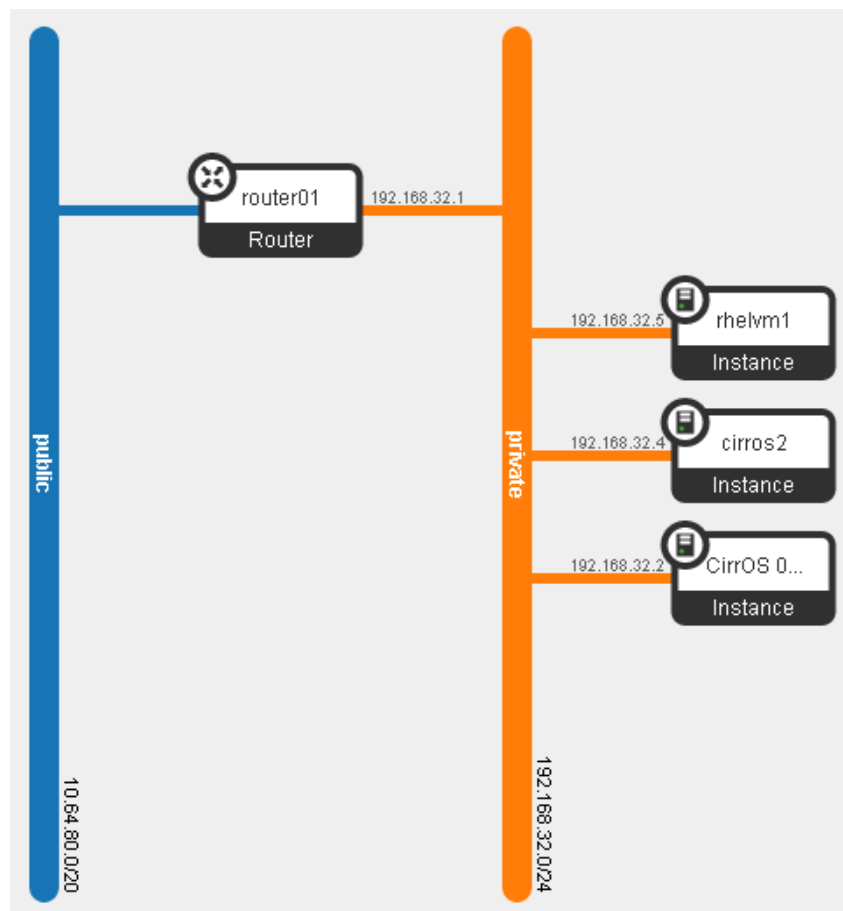
	Instance Name	Image Name	IP Address	Size	Keypair	Status	Task	Power State	Uptime	Actions
☐	rhelvm1	RHEL65	192.168.32.5 10.64.80.203	m1.small 2GB RAM 1 VCPU 20.0GB Disk	demokey	Active	None	Running	1 day, 1 hour	Create

Test the connectivity to the floating IP from the same client server.

```
[root@CR1-Mgmt1 ~]# ssh -i demokey.pem cloud-user@10.64.80.203 uptime
04:31:47 up 23:21,0 users,load average: 0.00, 0.00, 0.00
```

Create multiple instances to test the setup. After multiple instances are launched, the network topology will look as shown below.

Figure 27. Network topology



Volume management

Volumes are block devices that can be attached to instances. The HP 3PAR drivers for OpenStack cinder execute the volume operations by communicating with the HP 3PAR storage system over HTTP/HTTPS and SSH connections. Volumes are carved out from HP 3PAR StoreServ and presented to the instances. Use the dashboard to create and attach the volumes to the instances.

1. Log in to the dashboard as demo user. Navigate to Manage Compute → Volumes → Click on the + Create Volume button. Key in the volume name and required size. Click on the Create Volume button.

Figure 28. Create new volume

Create Volume

Volume Name *
data_vol

Description
Additional information here...

Type
▼

Size (GB) *
20

Volume Source
No source, empty volume. ▼

Description:
Volumes are block devices that can be attached to instances.

Volume Limits
Total Gigabytes (6 GB) 1,000 GB Available
Number of Volumes (1) 10 Available

Cancel Create Volume

2. Verify the creation on HP 3PAR Management Console. Note that there are no Hosts mappings shown in the lower part of the figure below.

Figure 29. 3PAR Virtual Volumes display

Provisioning : Domains : rhos_d0 : Virtual Volumes

Summary Virtual Volumes CPG Space Alerts

Summary 4 objects Filter Clear Any column contain

Name	System	Set	State	Type	Provisioning	RAID	Virtual Size (GiB)
rhos_glance_vv	CS700x-EVA	--	Normal	Base	Full	RAID 5	400.000
osv-5ngTrNfmRhIvbn9vGipQ (1)	CS700x-EVA	--	Normal	Base	Thin	RAID 5	6.000
osv-pmPw4Y.mQ5uKGK04w7fAlw	CS700x-EVA	--	Normal	Base	Thin	RAID 5	18.750
							430.750

Virtual Volume Details: osv-pmPw4Y.mQ5uKGK04w7fAlw

Summary Settings VLUNs Hosts Alerts

Summary 0 objects Filter Clear Any column contain

Name	System	Host Ports	Storage System Ports	Node ID(s)	Persona	Volumes Exported	Total Reserved Size (GiB)	Total Exported Size (GiB)

- From the dashboard, click on Edit Attachments for the volume data_vol that was newly created. This will pop-up a Manage Volume Attachments page to configure the instance to which this volume must be attached to. Choose the rhelvm1 instance that was created earlier and click on the Attach Volume button at the bottom. Once attached you can see the status on the dashboard.

Figure 30. Volumes status

Volumes

Logged in as: demo

Settings

He

Volumes

Filter

Filter

+ Create Volume

De

☐	Name	Description	Size	Status	Type	Attached To	Actions
☐	data_vol		20GB	In-Use	-	Attached to rhelvm1 on /dev/vdb	Edit Attachments

- Verify on HP 3PAR Management Console. You should now see the Hosts mappings populated. The volume will be presented to the compute node that hosts the rhelvm1 instance.

Figure 31. Volume Mapping to Host

Provisioning : Domains : rhos_d0 : Virtual Volumes									
Summary Virtual Volumes CPG Space Alerts									
Summary 4 objects Filter Clear Any column conta									
Name	System	Set	State	Type	Provisioning	RAID	Virtual Size (GiB)		
rhos_glance_vv	CS700x-EVA	--	Normal	Base	Full	RAID 5	400.00C		
osv-5ngTrNfMhIvbn9vIGpQ (1)	CS700x-EVA	--	Normal	Base	Thin	RAID 5	6.00C		
osv-pmPw4Y.mQ5uKGK04w7fAlw	CS700x-EVA	--	Normal	Base	Thin	RAID 5	18.75C		
							430.75C		
Virtual Volume Details: osv-pmPw4Y.mQ5uKGK04w7fAlw									
Summary Settings VLUNs Hosts Alerts									
Summary 1 object Filter Clear Any column cont									
Name	System	Host Ports	Storage System Ports	Node ID(s)	Persona	Volumes Exported	Total Reserved Size (GiB)	Total Exported Size (GiB)	
nova1	CS700x-EVA	2	4	0,1,2,3	1 - Generic	1	1.250	18.750	

- Verify from within the instance. Log in to the VM instance and run the fdisk command as shown below. The disk /dev/vdb is the newly attached volume.

```
[root@CR1-Mgmt1~(keystone_demo)]# ssh -i demokey.pem cloud-user@192.168.32.5
```

```
[cloud-user@rhelvm1 ~]$ sudo fdisk -l
```

```
Disk /dev/vda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000397ec
```

```
Device Boot      Start         End      Blocks   Id  System
/dev/vda1   *           1         1959      15728640   83   Linux
```

```

Disk /dev/vdb: 20.1 GB, 20132659200 bytes
16 heads, 63 sectors/track, 39009 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

```

6. At this point you can now partition the volume as needed, create a file system on it and mount it for use on the VM.

A. Create a filesystem on the disk:

```

[cloud-user@rhelvm1 ~]$ sudo mkfs.ext4 /dev/vdb

mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
1228800 inodes, 4915200 blocks
245760 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
150 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736,
1605632, 2654208,
    4096000

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

```

B. Create a mountpoint:

```

[cloud-user@rhelvm1 ~]$ sudo mkdir /DATA

```

C. Mount the disk on the mountpoint:

```

[cloud-user@rhelvm1 ~]$ sudo mount /dev/vdb /DATA

```

D. Verify the mountpoint:

```

[cloud-user@rhelvm1 ~]$ mount
/dev/vda1 on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs
(rw,rootcontext="system_u:object_r:tmpfs_t:s0")
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
/dev/vdb on /DATA type ext4 (rw)

```

Bill of materials

Note

Part numbers are at time of publication and subject to change. The bill of materials does not include complete support options or other rack and power requirements. If you have questions regarding ordering, please consult with your HP Reseller or HP Sales Representative for more details. hp.com/large/contact/enterprise/index.html

Table 7. Bill of materials HP ConvergedSystem 700x (727178-B21)

Quantity	Part number	Description
1	727178-B21	HP ConvergedSystem 700x

Implementing a proof-of-concept

As a matter of best practice for all deployments, HP recommends implementing a proof-of-concept using a test environment that matches as closely as possible the planned production environment. In this way, appropriate performance and scalability characterizations can be obtained. For help with a proof-of-concept, contact an HP Services representative (hp.com/large/contact/enterprise/index.html) or your HP partner.

Summary

After understanding and working through the steps we've described, you should have a working small cloud that is scalable through the addition of compute and network nodes. OpenStack is a complex suite of software and may be configured in many different ways. This reference architecture should provide a baseline for implementation and can serve as a functional environment for many workloads. We recommend the excellent documentation on the OpenStack website if you want to learn more about the individual components and architectural choices available to you when setting up and running OpenStack.

The HP ConvergedSystem 700x is an excellent platform for implementation of OpenStack. It provides powerful, dense compute and storage capabilities for this reference architecture; and the iLO management capability is indispensable in managing a small cluster of this kind.

Enjoy your OpenStack Cloud!

Appendix A: Packstack answer file

Below is the Packstack answer file used for this reference architecture. Texts in blue are non-default values that need to be keyed-in by the user. Refer to Table 2 and Table 4 for information on IP address and where OpenStack services run.

```
[general]

# Path to a Public key to install on servers. If a usable key has not
# been installed on the remote servers the user will be prompted for a
# password and this key will be installed so the password will not be
# required again
CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub

# Set to 'y' if you would like Packstack to install MySQL
CONFIG_MYSQL_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Image
# Service (Glance)
CONFIG_GLANCE_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Block
# Storage (Cinder)
CONFIG_CINDER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Compute
# (Nova)
CONFIG_NOVA_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Networking (Neutron)
CONFIG_NEUTRON_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Dashboard (Horizon)
CONFIG_HORIZON_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Object
# Storage (Swift)
CONFIG_SWIFT_INSTALL=n

# Set to 'y' if you would like Packstack to install OpenStack
# Metering (Ceilometer)
CONFIG_CEILOMETER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Orchestration (Heat)
CONFIG_HEAT_INSTALL=n

# Set to 'y' if you would like Packstack to install the OpenStack
# Client packages. An admin "rc" file will also be installed
CONFIG_CLIENT_INSTALL=y

# Comma separated list of NTP servers. Leave plain if Packstack
# should not install ntpd on instances.
CONFIG_NTP_SERVERS=<< LAB NTP Servers >>

# Set to 'y' if you would like Packstack to install Nagios to monitor
# OpenStack hosts
CONFIG_NAGIOS_INSTALL=n

# Comma separated list of servers to be excluded from installation in
# case you are running Packstack the second time with the same answer
# file and don't want Packstack to touch these servers. Leave plain if
# you don't need to exclude any server.
EXCLUDE_SERVERS=
```

```

# Set to 'y' if you want to run OpenStack services in debug mode.
# Otherwise set to 'n'.
CONFIG_DEBUG_MODE=n

# The IP address of the server on which to install MySQL
CONFIG_MYSQL_HOST=10.64.80.83

# Username for the MySQL admin user
CONFIG_MYSQL_USER=root

# Password for the MySQL admin user
CONFIG_MYSQL_PW=password

# The IP address of the server on which to install the QPID service
CONFIG_QPID_HOST=10.64.80.83

# Enable SSL for the QPID service
CONFIG_QPID_ENABLE_SSL=n

# Enable Authentication for the QPID service
CONFIG_QPID_ENABLE_AUTH=n

# The password for the NSS certificate database of the QPID service
CONFIG_QPID_NSS_CERTDB_PW=d360ceb3935848fcb17c37677973abla

# The port in which the QPID service listens to SSL connections
CONFIG_QPID_SSL_PORT=5671

# The filename of the certificate that the QPID service is going to
# use
CONFIG_QPID_SSL_CERT_FILE=/etc/pki/tls/certs/qpid_selfcert.pem

# The filename of the private key that the QPID service is going to
# use
CONFIG_QPID_SSL_KEY_FILE=/etc/pki/tls/private/qpid_selfkey.pem

# Auto Generates self signed SSL certificate and key
CONFIG_QPID_SSL_SELF_SIGNED=y

# User for qpid authentication
CONFIG_QPID_AUTH_USER=qpid_user

# Password for user authentication
CONFIG_QPID_AUTH_PASSWORD=48ced586f0424847

# The IP address of the server on which to install Keystone
CONFIG_KEYSTONE_HOST=10.64.80.83

# The password to use for the Keystone to access DB
CONFIG_KEYSTONE_DB_PW=2a424cbf2da44ed2

# The token to use for the Keystone service api
CONFIG_KEYSTONE_ADMIN_TOKEN=6f15c3833eec427ba7f9ec063697dbc1

# The password to use for the Keystone admin user
CONFIG_KEYSTONE_ADMIN_PW=password

# The password to use for the Keystone demo user
CONFIG_KEYSTONE_DEMO_PW=password

# Keystone token format. Use either UUID or PKI
CONFIG_KEYSTONE_TOKEN_FORMAT=PKI

# The IP address of the server on which to install Glance
CONFIG_GLANCE_HOST=10.64.80.83

# The password to use for the Glance to access DB

```

```

CONFIG_GLANCE_DB_PW=13c1491c952d4999

# The password to use for the Glance to authenticate with Keystone
CONFIG_GLANCE_KS_PW=651ed00480764368

# The IP address of the server on which to install Cinder
CONFIG_CINDER_HOST=10.64.80.83

# The password to use for the Cinder to access DB
CONFIG_CINDER_DB_PW=f17f20afbef642bc

# The password to use for the Cinder to authenticate with Keystone
CONFIG_CINDER_KS_PW=c2b8543fafcd44bb

# The Cinder backend to use, valid options are: lvm, gluster, nfs
CONFIG_CINDER_BACKEND=lvm

# Create Cinder's volumes group. This should only be done for testing
# on a proof-of-concept installation of Cinder. This will create a
# file-backed volume group and is not suitable for production usage.
CONFIG_CINDER_VOLUMES_CREATE=y

# Cinder's volumes group size. Note that actual volume size will be
# extended with 3% more space for VG metadata.
CONFIG_CINDER_VOLUMES_SIZE=20G

# A single or comma separated list of gluster volume shares to mount,
# eg: ip-address:/vol-name, domain:/vol-name
CONFIG_CINDER_GLUSTER_MOUNTS=

# A single or comma separated list of NFS exports to mount, eg: ip-
# address:/export-name
CONFIG_CINDER_NFS_MOUNTS=

# The IP address of the server on which to install the Nova API
# service
CONFIG_NOVA_API_HOST=10.64.80.83

# The IP address of the server on which to install the Nova Cert
# service
CONFIG_NOVA_CERT_HOST=10.64.80.83

# The IP address of the server on which to install the Nova VNC proxy
CONFIG_NOVA_VNCPROXY_HOST=10.64.80.83

# A comma separated list of IP addresses on which to install the Nova
# Compute services
CONFIG_NOVA_COMPUTE_HOSTS=10.64.80.85,10.64.80.86,10.64.80.87,10.64.80.88,10.64.80
.89,10.64.80.90

# The IP address of the server on which to install the Nova Conductor
# service
CONFIG_NOVA_CONDUCTOR_HOST=10.64.80.83

# The password to use for the Nova to access DB
CONFIG_NOVA_DB_PW=0c7320b764ba45f5

# The password to use for the Nova to authenticate with Keystone
CONFIG_NOVA_KS_PW=ce33e3d3af7b49bf

# The IP address of the server on which to install the Nova Scheduler
# service
CONFIG_NOVA_SCHED_HOST=10.64.80.83

# The overcommitment ratio for virtual to physical CPUs. Set to 1.0
# to disable CPU overcommitment

```

```

CONFIG_NOVA_SCHED_CPU_ALLOC_RATIO=16.0

# The overcommitment ratio for virtual to physical RAM. Set to 1.0 to
# disable RAM overcommitment
CONFIG_NOVA_SCHED_RAM_ALLOC_RATIO=1.5

# Private interface for Flat DHCP on the Nova compute servers
CONFIG_NOVA_COMPUTE_PRIVIF=eth1

# The list of IP addresses of the server on which to install the Nova
# Network service
CONFIG_NOVA_NETWORK_HOSTS=10.64.80.81

# Nova network manager
CONFIG_NOVA_NETWORK_MANAGER=nova.network.manager.FlatDHCPManager

# Public interface on the Nova network server
CONFIG_NOVA_NETWORK_PUBIF=eth0

# Private interface for network manager on the Nova network server
CONFIG_NOVA_NETWORK_PRIVIF=eth1

# IP Range for network manager
CONFIG_NOVA_NETWORK_FIXEDRANGE=192.168.32.0/22

# IP Range for Floating IP's
CONFIG_NOVA_NETWORK_FLOATRANGE=10.3.4.0/22

# Name of the default floating pool to which the specified floating
# ranges are added to
CONFIG_NOVA_NETWORK_DEFAULTFLOATINGPOOL=nova

# Automatically assign a floating IP to new instances
CONFIG_NOVA_NETWORK_AUTOASSIGNFLOATINGIP=n

# First VLAN for private networks
CONFIG_NOVA_NETWORK_VLAN_START=100

# Number of networks to support
CONFIG_NOVA_NETWORK_NUMBER=1

# Number of addresses in each private subnet
CONFIG_NOVA_NETWORK_SIZE=255

# The IP addresses of the server on which to install the Neutron
# server
CONFIG_NEUTRON_SERVER_HOST=10.64.80.83

# The password to use for Neutron to authenticate with Keystone
CONFIG_NEUTRON_KS_PW=5d99f9e9b7f743fe

# The password to use for Neutron to access DB
CONFIG_NEUTRON_DB_PW=a93ec865b87c4d57

# A comma separated list of IP addresses on which to install Neutron
# L3 agent
CONFIG_NEUTRON_L3_HOSTS=10.64.80.84

# The name of the bridge that the Neutron L3 agent will use for
# external traffic, or 'provider' if using provider networks
CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex

# A comma separated list of IP addresses on which to install Neutron
# DHCP agent
CONFIG_NEUTRON_DHCP_HOSTS=10.64.80.84

# A comma separated list of IP addresses on which to install Neutron

```

```

# LBaaS agent
CONFIG_NEUTRON_LBAAS_HOSTS=10.64.80.84

# The name of the L2 plugin to be used with Neutron
CONFIG_NEUTRON_L2_PLUGIN=openvswitch

# A comma separated list of IP addresses on which to install Neutron
# metadata agent
CONFIG_NEUTRON_METADATA_HOSTS=10.64.80.84

# A comma separated list of IP addresses on which to install Neutron
# metadata agent
CONFIG_NEUTRON_METADATA_PW=f2d1b874d5184121

# A comma separated list of network type driver entrypoints to be
# loaded from the neutron.ml2.type_drivers namespace.
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=local

# A comma separated ordered list of network_types to allocate as
# tenant networks. The value 'local' is only useful for single-box
# testing but provides no connectivity between hosts.
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=local

# A comma separated ordered list of networking mechanism driver
# entrypoints to be loaded from the neutron.ml2.mechanism_drivers
# namespace.
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=openvswitch

# A comma separated list of physical_network names with which flat
# networks can be created. Use * to allow flat networks with arbitrary
# physical_network names.
CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*

# A comma separated list of <physical_network>:<vlan_min>:<vlan_max>
# or <physical_network> specifying physical_network names usable for
# VLAN provider and tenant networks, as well as ranges of VLAN tags on
# each available for allocation to tenant networks.
CONFIG_NEUTRON_ML2_VLAN_RANGES=

# A comma separated list of <tun_min>:<tun_max> tuples enumerating
# ranges of GRE tunnel IDs that are available for tenant network
# allocation. Should be an array with tun_max +1 - tun_min > 1000000
CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=

# Multicast group for VXLAN. If unset, disables VXLAN enable sending
# allocate broadcast traffic to this multicast group. When left
# unconfigured, will disable multicast VXLAN mode. Should be an
# Multicast IP (v4 or v6) address.
CONFIG_NEUTRON_ML2_VXLAN_GROUP=

# A comma separated list of <vni_min>:<vni_max> tuples enumerating
# ranges of VXLAN VNI IDs that are available for tenant network
# allocation. Min value is 0 and Max value is 16777215.
CONFIG_NEUTRON_ML2_VNI_RANGES=

# The name of the L2 agent to be used with Neutron
CONFIG_NEUTRON_L2_AGENT=openvswitch

# The type of network to allocate for tenant networks (eg. vlan,
# local)
CONFIG_NEUTRON_LB_TENANT_NETWORK_TYPE=local

# A comma separated list of VLAN ranges for the Neutron linuxbridge
# plugin (eg. physnet1:1:4094,physnet2,physnet3:3000:3999)
CONFIG_NEUTRON_LB_VLAN_RANGES=

```

```

# A comma separated list of interface mappings for the Neutron
# linuxbridge plugin (eg. physnet1:br-eth1,physnet2:br-eth2,physnet3
# :br-eth3)
CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS=

# Type of network to allocate for tenant networks (eg. vlan, local,
# gre, vxlan)
CONFIG_NEUTRON_OVS_TENANT_NETWORK_TYPE=vlan

# A comma separated list of VLAN ranges for the Neutron openvswitch
# plugin (eg. physnet1:1:4094,physnet2,physnet3:3000:3999)
CONFIG_NEUTRON_OVS_VLAN_RANGES=physnet1:1000:1050

# A comma separated list of bridge mappings for the Neutron
# openvswitch plugin (eg. physnet1:br-eth1,physnet2:br-eth2,physnet3
# :br-eth3)
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=physnet1:br-eth1

# A comma separated list of colon-separated OVS bridge:interface
# pairs. The interface will be added to the associated bridge.
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=br-eth1:eth1

# A comma separated list of tunnel ranges for the Neutron openvswitch
# plugin (eg. 1:1000)
CONFIG_NEUTRON_OVS_TUNNEL_RANGES=

# The interface for the OVS tunnel. Packstack will override the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface. (eg. eth1)
CONFIG_NEUTRON_OVS_TUNNEL_IF=

# VXLAN UDP port
CONFIG_NEUTRON_OVS_VXLAN_UDP_PORT=4789

# The IP address of the server on which to install the OpenStack
# client packages. An admin "rc" file will also be installed
CONFIG_OSCLIENT_HOST=10.64.80.81

# The IP address of the server on which to install Horizon
CONFIG_HORIZON_HOST=10.64.80.81

# To set up Horizon communication over https set this to "y"
CONFIG_HORIZON_SSL=n

# PEM encoded certificate to be used for ssl on the https server,
# leave blank if one should be generated, this certificate should not
# require a passphrase
CONFIG_SSL_CERT=

# Keyfile corresponding to the certificate if one was entered
CONFIG_SSL_KEY=

# The IP address on which to install the Swift proxy service
# (currently only single proxy is supported)
CONFIG_SWIFT_PROXY_HOSTS=10.64.80.81

# The password to use for the Swift to authenticate with Keystone
CONFIG_SWIFT_KS_PW=2e069453b8684a25

# A comma separated list of IP addresses on which to install the
# Swift Storage services, each entry should take the format
# <ipaddress>[/dev], for example 127.0.0.1/vdb will install /dev/vdb
# on 127.0.0.1 as a swift storage device(packstack does not create the
# filesystem, you must do this first), if /dev is omitted Packstack
# will create a loopback device for a test setup
CONFIG_SWIFT_STORAGE_HOSTS=10.64.80.81

```

```
# Number of swift storage zones, this number MUST be no bigger than
# the number of storage devices configured
CONFIG_SWIFT_STORAGE_ZONES=1

# Number of swift storage replicas, this number MUST be no bigger
# than the number of storage zones configured
CONFIG_SWIFT_STORAGE_REPLICAS=1

# FileSystem type for storage nodes
CONFIG_SWIFT_STORAGE_FSTYPE=ext4

# Shared secret for Swift
CONFIG_SWIFT_HASH=675c7037b1014732

# Size of the swift loopback file storage device
CONFIG_SWIFT_STORAGE_SIZE=2G

# Whether to provision for demo usage and testing
CONFIG_PROVISION_DEMO=n

# The CIDR network address for the floating IP subnet
CONFIG_PROVISION_DEMO_FLOATRANGE=172.24.4.224/28

# Whether to configure tempest for testing
CONFIG_PROVISION_TEMPEST=n

# The uri of the tempest git repository to use
CONFIG_PROVISION_TEMPEST_REPO_URI=https://github.com/openstack/tempest.git

# The revision of the tempest git repository to use
CONFIG_PROVISION_TEMPEST_REPO_REVISION=stable/havana

# Whether to configure the ovs external bridge in an all-in-one
# deployment
CONFIG_PROVISION_ALL_IN_ONE_OVS_BRIDGE=n

# The IP address of the server on which to install Heat service
CONFIG_HEAT_HOST=10.64.80.81

# The password used by Heat user to authenticate against MySQL
CONFIG_HEAT_DB_PW=81c604c6f49f4d62

# The password to use for the Heat to authenticate with Keystone
CONFIG_HEAT_KS_PW=44c4b62c77454a37

# Set to 'y' if you would like Packstack to install Heat CloudWatch
# API
CONFIG_HEAT_CLOUDWATCH_INSTALL=n

# Set to 'y' if you would like Packstack to install Heat
# CloudFormation API
CONFIG_HEAT_CFN_INSTALL=n

# The IP address of the server on which to install Heat CloudWatch
# API service
CONFIG_HEAT_CLOUDWATCH_HOST=10.64.80.81

# The IP address of the server on which to install Heat
# CloudFormation API service
CONFIG_HEAT_CFN_HOST=10.64.80.81

# The IP address of the server on which to install Ceilometer
CONFIG_CEILOMETER_HOST=10.64.80.81

# Secret key for signing metering messages.
CONFIG_CEILOMETER_SECRET=f92dee35bbce4220
```

```
# The password to use for Ceilometer to authenticate with Keystone
CONFIG_CEILOMETER_KS_PW=700a59f3fbff41f1

# The IP address of the server on which to install the Nagios server
CONFIG_NAGIOS_HOST=10.64.80.81

# The password of the nagiosadmin user on the Nagios server
CONFIG_NAGIOS_PW=70cf5a0d1d9f4369

# To subscribe each server to EPEL enter "y"
CONFIG_USE_EPEL=n

# A comma separated list of URLs to any additional yum repositories
# to install
CONFIG_REPO=

# To subscribe each server with Red Hat subscription manager, include
# this with CONFIG_RH_PW
CONFIG_RH_USER=

# To subscribe each server with Red Hat subscription manager, include
# this with CONFIG_RH_USER
CONFIG_RH_PW=

# To subscribe each server to Red Hat Enterprise Linux 6 Server Beta
# channel (only needed for Preview versions of RHOS) enter "y"
CONFIG_RH_BETA_REPO=n

# To subscribe each server with RHN Satellite, fill Satellite's URL
# here. Note that either satellite's username/password or activation
# key has to be provided
CONFIG_SATELLITE_URL=

# Username to access RHN Satellite
CONFIG_SATELLITE_USER=

# Password to access RHN Satellite
CONFIG_SATELLITE_PW=

# Activation key for subscription to RHN Satellite
CONFIG_SATELLITE_AKEY=

# Specify a path or URL to a SSL CA certificate to use
CONFIG_SATELLITE_CACERT=

# If required specify the profile name that should be used as an
# identifier for the system in RHN Satellite
CONFIG_SATELLITE_PROFILE=

# Comma separated list of flags passed to rhnreg_ks. Valid flags are:
# novirtinfo, norhnsd, nopackages
CONFIG_SATELLITE_FLAGS=

# Specify a HTTP proxy to use with RHN Satellite
CONFIG_SATELLITE_PROXY=

# Specify a username to use with an authenticated HTTP proxy
CONFIG_SATELLITE_PROXY_USER=

# Specify a password to use with an authenticated HTTP proxy.
CONFIG_SATELLITE_PROXY_PW=
```

Appendix B: Troubleshooting

1. Problem: Unable to reach private IP of a VM instance.

Solution: From the neutron server try to ping the VM private IP via the qrouter namespace using the commands below.

```
$ ip netns
qrouter-71e12c86-97d9-4dd7-9765-6cd584385916
qdhcp-98b541d2-33e4-4e2a-9bad-3624b6326965
```

```
$ ip netns exec qrouter-71e12c86-97d9-4dd7-9765-6cd584385916 ping -c 2 <VM IP>
```

Check the security group rules assigned to the VM instance. Verify that rules allow ICMP and SSH protocols. Enable all protocols from all networks for troubleshooting purposes.

If VM IP is unreachable, ping the private gateway IP:

```
$ ip netns exec qrouter-71e12c86-97d9-4dd7-9765-6cd584385916 ping -c 2 <Gateway IP>
```

If the Gateway IP is also not reachable, verify VLAN configuration starting from the Virtual Connect server profiles, Ethernet profiles and switch configurations.

2. Problem: Unable to reach the floating IP of a VM instance.

Solution: Follow a similar approach as described above. First try to ping the IP via the qrouter namespace. If negative, then try to ping the router's external gateway IP. If still not reachable, then verify VLAN configuration.

3. Problem: Unable to attach a volume to an instance. The /var/log/cinder/cinder.log shows an error – KeyError: 'wwpns'.

Solution: Possible cause is sysfstools and sg3-utils packages are not installed on the compute node. Install these packages and try to attach the volume again.

Portions of this white paper are used with permission from Red Hat, namely; "Deploying and Using Red Hat Enterprise Linux OpenStack Platform 3" by Jacob Liberman, Principal Software Engineer and "Red Hat Enterprise Linux OpenStack Platform 4 – Getting Started Guide"

WARRANTY DISCLAIMER

HP MAKES NO EXPRESS OR IMPLIED WARRANTY OF ANY KIND REGARDING THE SYSTEM AND SOFTWARE DESCRIBED IN THIS WHITE PAPER, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT. HP SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL THEORY, IN CONNECTION WITH OR ARISING OUT OF THE FURNISHING, PERFORMANCE OR USE OF THE SYSTEM AND SOFTWARE DESCRIBED IN THIS WHITE PAPER.

For more information

Red Hat Enterprise Linux OpenStack Platform:

https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform

OpenStack® HP 3PAR StoreServ Block Storage Drivers Configuration Best Practices:

<http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA4-9853ENW>

OpenStack foundation documents:

<http://docs.OpenStack.org>

HP ConvergedSystem 700x, hp.com/go/convergedsystem/cs700x

To help us improve our documents, please provide feedback at hp.com/solutions/feedback.

Sign up for updates

hp.com/go/getupdated

© Copyright 2014 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation. AMD is a trademark of Advanced Micro Devices, Inc. Intel is a trademark of Intel Corporation in the U.S. and other countries. Red Hat and Red Hat Enterprise Linux are registered trademarks of Red Hat, Inc. in the United States and other countries.

The OpenStack Word Mark and OpenStack Logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed, or sponsored by the OpenStack Foundation, or the OpenStack community.

